EBERHARD-KARLS-UNIVERSITÄT TÜBINGEN
ZENTRUM FÜR DATENVERARBEITUNG

Ralf Banning · Heiko A. Groeneveld

Textsatz und Layout mit LaTeX

SCHRIFTENREIHE DES ZDV Nr. 3

Textsatz und Layout

mit

LaTeX

von

Ralf Banning
Heiko A. Groeneveld

Online-Version (Version ONL.300-1.95) für 300 dpi Drucker

`http://www.uni-tuebingen.de/zdv/zrainfo/sw/text/l2e-ONL.300-1.95.ps.gz`

Satz: Ralf Banning und Heiko A. Groeneveld.
Gesetzt mit LaTeX unter Verwendung der Computer Modern Schriften.

Das Titelbild zeigt ein Porträt von Johannes Gutenberg.

Vorwort

Dieses Buch ist auf Grundlage eines Kurses entstanden, der durch das Zentrum für Datenverarbeitung (ZDV) in den letzten Jahren regelmäßig an der Universität Tübingen angeboten wurde. Mit der Herausgabe von LaTeX 2ε im Jahre 1994 ergab sich die Notwendigkeit, eine völlig überarbeitete Fassung der Kursunterlagen herauszubringen. Wir haben die Gelegenheit genutzt und ein grundlegend neues Konzept ausgearbeitet, das nun in diesem Buch vorliegt.

Nach wie vor ist dieses Buch als Begleitmaterial für unsere Kursteilnehmer gedacht. Es enthält daher alle Originalfolien, die auch im Kurs Verwendung finden. Die Folien sind so aufgebaut, daß der Benutzer in kleinen, thematisch an den Aufgaben des Textsatzes orientierten Einheiten, an die Strukturen des LaTeX-Programms herangeführt wird. Mit dem vorliegenden Material ist der Benutzer am Ende in der Lage, auch komplizierte Textsatzaufgaben, wie mehrspaltigen Satz oder Formelsatz, zu bewältigen. In diesem Sinne vermittelt der „Folienteil“ das *Know How* für den Textsatz mit LaTeX.

Neu sind die Ergänzungsseiten. Wir haben sie einerseits in das Buch aufgenommen, um den Kursteilnehmern auch solche Informationen schriftlich an die Hand zu geben, die während der Kurse lediglich vorgetragen werden. Andererseits ergab sich hierdurch die Möglichkeit, auf Hintergründe des LaTeX-Programms einzugehen, ohne die Ausbildung zu überfrachten. Die Ergänzungsseiten sind daher weitestgehend abgeschlossene Darstellungen, die sich zum „Schmökern“ eignen und die Kenntnisse über das Programm vertiefen sollen. Insbesondere haben wir viel Wert auf die Darstellung der traditionellen Kunst der Typographie gelegt, die nach wie vor Grundlage jeden guten Textsatzes ist, und in vielen Darstellungen über den Computersatz zu kurz kommt. Die Ergänzungsseiten liefern dem Benutzer in diesem Sinne zusätzliches *Know Why* für den Umgang mit LaTeX.

Das LaTeX-Programm befindet sich weiter in Entwicklung. LaTeX verdankt seine Vielseitigkeit insbesondere den Ergänzungen, die von vielen Benutzern in der ganzen Welt beigetragen werden. Soweit sich diese Ergänzungen inzwischen als Standard etabliert haben, werden sie in diesem Buch beschrieben. Darüberhinaus haben wir auch einige weit verbreitete und am ZDV installierte Ergänzungen erklärt. Trotzdem blieben im Interesse der Überschaubarkeit einige interessante Entwicklungen unberücksichtigt, zum Beispiel der Textsatz mit nicht lateinischen Alphabeten. Der interessierte Leser findet ausführliche Informationen zu LaTeX-Erweiterungen z.B. in [4] und [9].

Schließlich haben wir uns um eine reichhaltige Ausstattung des Buches mit typischen Anwendungsbeispielen bemüht, um das Erfassen der Materie zu erleichtern. In der Regel ist der Eingabeteil eines Beispiels grau hinterlegt, die zugehörige Ausgabe in einem schattierten Rahmen wiedergegeben.

Im Sommer 1995

Die Autoren

Inhaltsverzeichnis

Abschnitt 1

Programmstruktur von LaTeX

Seit Aufkommen des elektronischen Textsatzes wird zunehmend die Funktion des Autors und des Schriftsetzers in einer Person vereinigt. Daher besteht für jeden Autor, der ein Textsatzprogramm zur Abfassung seiner Schriftstücke benutzt, die Notwendigkeit, sich mit den Konventionen und Regeln des Textsatzes auseinanderzusetzen.

Das Textsatzsystem LaTeX ist auf Grundlage typographischer Regeln entwickelt worden und stellt dem Benutzer ein umfassendes Werkzeug für einen typographisch sauberen Textsatz zur Verfügung, ohne daß sich dieser zu sehr um Detailfragen kümmern muß.

Wie bei jedem Werkzeug ist es allerdings sinnvoll, nicht nur zu wissen *wie* man es anwendet, sondern auch *warum* man es auf die jeweilige Art und Weise anwendet. Im folgenden Kapitel soll deshalb das Verhältnis des LaTeX Programmsystems zu den Regeln der Typographie erläutert werden.

1.1 Was ist Textsatz?

1.1.1

Was ist LaTeX?

- TeX (sprich Tech) ist ein von Donald E. Knuth an der Stanford University entwickeltes Textsatzsystem. Ursprünglich wurde es entwickelt, um insbesondere den Satz mathematischer Formeln zu unterstützen, doch inzwischen gelangt TeX in allen Bereichen des Textsatzes zur Anwedung, z.B. für den Textsatz mit nicht-lateinischen Alphabeten (Koreanisch, Arabisch, Hebräisch). Die typographische Qualität ist mit hochwertigem Buchdrucksatz vergleichbar.
- LaTeX ist ein ursprünglich von Leslie Lamport entwickeltes Makropaket für TeX, welches eine Reihe verständlicher Befehle bereitstellt, die sich an der logischen Struktur von Textdokumenten orientieren. Beschränkt sich der Anwender auf diese Befehle, so bleibt er von den Details der drucktechnischen Gestaltung weitestgehend verschont.
- Implementationen von TeX und LaTeX gibt es für die verschiedensten Rechnerarchitekturen — vom PC bis zum Großrechner, von DOS bis UNIX. Sowohl die Handhabung der Programme als auch deren Ergebnisse sind als vergleichsweise plattformunabhängig anzusehen.

Nur ... was ist eigentlich Textsatz?

- Unter *Setzen* versteht man das Plazieren eines in einem allgemeinen Sinne „graphischen“ Elementes auf einer Seite. Der Satz hat dabei einem *Layout* zu folgen, welches Vorschriften für die Anordnung der jeweiligen Elemente macht.
- Textsatz befaßt sich demzufolge mit dem Setzen von Text, also dem „Zusammensetzen“ von Buchstaben. Das graphische Element, dessen sich der Setzer dabei bedient, ist somit jeweils ein einzelner Buchstabe, ein Satz- oder ein Sonderzeichen.

Weitere Anmerkungen zu LaTeX

TeX und LaTeX. Als Donald Knuth 1983 eine stabile Version von TeX herausbrachte, konnte er bereits auf eine gut zehnjährige Entwicklungszeit zurückblicken. Der Programmname wurde schon für die erste Version „TeX78“ verwendet und spielt auf das griechische Wort $\tau\epsilon\chi\nu\eta$ an, das sowohl „Kunst“ als auch „Technologie“ bedeutet. Diese Doppelbedeutung entsprach genau seiner Vorstellung, was TeX zu leisten imstande sein sollte: technologisch fortgeschrittenen Textsatz auf höchstem typographischen Niveau. Knuth selbst bemerkte zur eigentümlichen Aussprache von TeX [6]:

> Insiders pronounce the χ of TeX as a Greek chi, not as an ‘x’, so that TeX rhymes with the word blechhh. It’s the ‘ch’ sound in Scottish words like *loch* or German words like *ach*; it’s the Spanish ‘j’ and a Russian ‘kh’. When you say it correctly to your computer, the terminal may become slightly moist.

Als universelles Satzsystem entwickelt, eröffnet TeX alle Möglichkeiten des Textsatzes, freilich mit dem Nachteil, daß der Benutzer einem ungewöhnlich großen Sprachumfang von etwa 900 Befehlen gegenübersteht (Man vergleiche dies mit dem Befehlssatz von C oder Fortran Compilern!). Etwa 600 dieser Befehle sind aus 300 fest einkompilierten Befehlen („Intrinsics“) als Makros abgeleitet. Genau an dieser Stelle hat Leslie Lamport angesetzt und eine Makrosammlung (nämlich LaTeX) zusammengestellt, die für die meisten immer wiederkehrenden Satzprobleme standardisierte Lösungen anbietet [12].

LaTeX2.09 und LaTeX 2$_\varepsilon$. Im Jahr 1994 wurde die seit 1986 aktuelle Version von LaTeX (im folgenden LaTeX2.09 genannt) durch LaTeX 2$_\varepsilon$ abgelöst. Die Unterschiede von „alt“ zu „neu“ sind gering. LaTeX2.09 Quellen können auch weiterhin mit LaTeX 2$_\varepsilon$ verarbeitet werden. Das vorliegende Buch beschreibt die Eigenschaften von LaTeX 2$_\varepsilon$, das wir deshalb im folgenden einfach LaTeX nennen werden (siehe auch [2]).

LaTeX am zdv. Das Tübinger Zentrum für Datenverarbeitung (ZDV) stellt das LaTeX-System (einschließlich vieler Erweiterungen und allen standardisierten Zusatzprogrammen) auf zentralen Servern und den Workstation-Pools zur Verfügung.

Des weiteren bietet das ZDV regelmäßige Beratungstermine für LaTeX-Benutzer an, sowie Kompaktkurse, um den Einstieg in den Textsatz mit LaTeX zu erleichtern.

Für Benutzer mit eigenem Rechner bietet das ZDV installationsfertige LaTeX-Pakete für die folgenden Betriebssysteme an (Stand August ’95):

UNIX-Plattformen:

- HP-UX
- AIX
- IRIX

PC’s:

- MS-DOS

Über den aktuellen Stand des Angebots kann man sich über das WWW informieren (Homepage des ZDV):

`http://www.uni-tuebingen.de/zdv/zdv-home.html`

Elektronischer Textsatz und die Boxenlogik von TEX

- Beim *elektronischen* Textsatz mit TEX werden die Buchstaben durch eine *Box* mit jeweils bestimmter *Höhe*, *Tiefe* und *Breite* repräsentiert (Abbildung 1.1). Entlang der *Grundlinie* werden diese Boxen wie Perlen an einer Schnur zu Wörtern aufgereit.

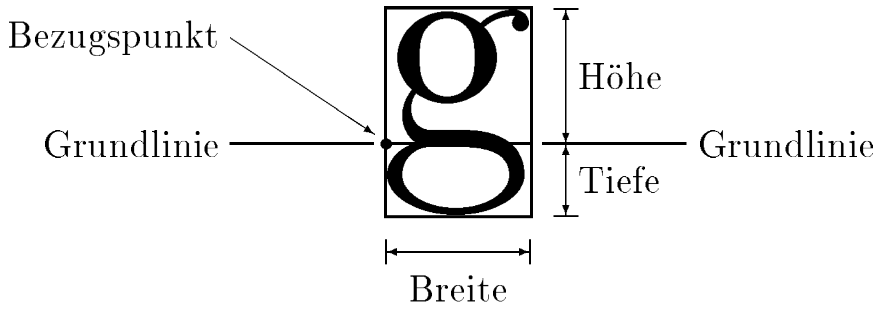


Abbildung 1.1: Geometrie eines Buchstabens

- Beim elektronischen Textsatz enthält die Box eines Zeichens oft auch Information über den Abstand zum nächsten Zeichen. TEX setzt tatsächlich nicht die Zeichen selbst, sondern nur diese Boxen, indem es mit deren Größen (Höhe, Breite, Tiefe, Anschlußwerte) „rechnet".

- Worte werden in Sätzen (zunächst) durch den *natürlichen Wortabstand* voneinander getrennt, welcher Bestandteil des jeweiligen Zeichensatzes ist. In den meisten Sprachen schreibt und setzt man von links nach rechts, bis der *Spaltenrand* erreicht ist.

- Am jeweiligen Zeilenende wird der *Zeilenumbruch* vorgenommen, d.h. die überschüssigen, nachfolgenden Wörter werden in die nächste Zeile übernommen. Diese Zeile wird dann wieder bis zum Zeilenende aufgefüllt u.s.w.

- Es wird Zeile um Zeile umgebrochen und vertikal an die jeweils vorherige gereiht (wobei der Zeilenabstand vom Layout bestimmt wird, auch, wenn der jeweilige Zeichensatz einen Anhalt, eine Art „natürlichen Zeilenabstand" liefert), bis ein vollständiger *Absatz*[1] gesetzt ist.

[1] Absätze werden im Satzgewerbe gelegentlich auch als Paragraphen bezeichnet.

Typographie – Eine Vorgeschichte

Die Typographie befaßt sich mit der Gestaltung eines Textes durch Schriften und andere Zeichen. Wörtlich übersetzt bedeutet *Typographie* „Schreiben mit Typen“. Dies wird verständlich, wenn man bedenkt, daß im klassischen Buchdruck (Bleisatz) jeder Buchstabe spiegelverkehrt auf einem Stempel erhaben ausgebildet ist und *Type* oder auch *Letter* genannt wird (Abb. 1.2). Aus den Lettern entstehen durch Aneinanderreihung Wörter. Durch die typographische Gestaltung soll der Leser bei der Aufnahme des Textinhalts unterstützt werden. Dies kann z.B. durch die geeignete Wahl der Schriftart, der Größe der Buchstaben und die Verteilung des Textes auf der Seite erreicht werden. Bei der Wahl der Gestaltungsmittel kommt es also weniger darauf an, einen „schönen“ Textsatz zu erzielen, sondern vielmehr die Lesbarkeit des Textes zu erhöhen. Guter Textsatz ist im typographischen Sinne also formal unauffällig. Die Typographie wird daher auch eine *unsichtbare Kunst* genannt.

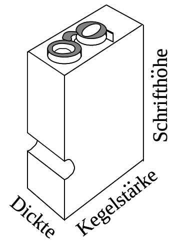


Abbildung 1.2: Letter

Die Fachbegriffe der Typographie orientieren sich auch heute noch stark an der Arbeitsweise des Handsatzes. Hierbei werden die Lettern auf einem Winkelhaken gesetzt (Abb. 1.3). Jede Letter hat einen Bezugspunkt für die Grundlinie und eine bestimmte Breite (Dickte). Damit auch Buchstaben mit Unterlängen (wie z.B der Buchstabe „g“) die Grundlinie halten können, müssen beim Bleisatz alle Lettern einer Schrift die gleiche Kegelstärke besitzen. Der Bezugspunkt befindet sich also in diesem Fall an der Unterkante der Letter. Beim elektronischem Satz ist die Letter durch eine „logische“ Box ersetzt, die die Maße der Letter wiedergibt. Eine gleichmäßige Kegelstärke ist nicht mehr erforderlich. Stattdessen befindet sich heute der Bezugspunkt auf der Grundlinie (Abbildung 1.1). Die Grundlinie teilt hierbei die Letter in ihren Korpus (*Höhe* über der Grundlinie) und ihre Unterlängen (*Tiefe* unter der Grundlinie; Abb. 1.1).

Als Begründer des Buchdrucks mit beweglichen Lettern gilt Johannes Gutenberg (*1400?, †1468?). Hergestellt wurden Bücher natürlich schon viel früher. Dies waren zum einen Handschriften und später Bücher, die mit Holzdruckplatten gedruckt wurden. Bei dieser Technik mußte für jede Seite ein eigener — rasch verschlissener — Druckstock geschnitten werden. Das war mühsam und teuer. Das Zusammensetzen des Druckstocks aus einzelnen, „frei beweglichen“ Lettern war dagegen eine vergleichsweise einfache und schnell zu erledigende Tätigkeit. Zudem konnten die Lettern nach dem Druck wiederverwendet und zu neuen Druckstöcken zusammengesetzt werden. Gutenberg war der erste, der ein großes Buch (die sog. Gutenberg Bibel) in dieser neuen Technik gedruckt hat (um 1450). Er hat mit der Gestaltung seiner zweispaltigen, 42-zeiligen Bibel bis heute die Typographie wesentlich beeinflußt.

Der Handsatz wurde seit dem 19. Jahrhundert durch den Maschinensatz abgelöst, bei dem die Lettern zeilenweise in sog. Matrizen gegossen wurden (Linotype, Typograph, Intertype, ...)

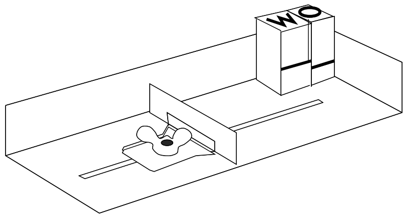


Abbildung 1.3: Winkelhaken

Heute wird Textsatz vorwiegend als sog. Computersatz durchgeführt; hierfür ist LaTeX ein Beispiel.

1.1.3 Absatzgestaltung

Von Kunstformen einmal abgesehen, gibt es zwei grundlegende Möglichkeiten des Absatzlayouts. Beim *Flattersatz* werden nach dem Zeilenumbruch die Worte in ihrem natürlichen Abstand belassen, so daß aufgrund der von Zeile zu Zeile unterschiedlichen Ausnutzung der Spaltenbreite ein ungleichmäßiger (flatternder) rechter Absatzrand entsteht.

linker Spaltenrand	Beim Flattersatz werden die Zeilen solange gefüllt, wie es die Spaltenbreite zuläßt. Sobald ein Wort nicht mehr in eine Zeile paßt, wird es in die nächste Zeile ge- und der Satz in derselben fortgesetzt.	rechter Spaltenrand

Werden die Wortabstände so verändert (Austreiben oder Einbringen), daß ein glatter linker und rechter Zeilenrand entsteht, so spricht man von *Blocksatz*.

linker Spaltenrand	Beim Blocksatz werden die Zeilen ebenfalls solange gefüllt, wie es die Spaltenbreite zuläßt. Anschließend wird die Zeilenlänge durch Austreiben oder Einbringen auf das Maß der Spaltenbreite gebracht.	rechter Spaltenrand

Die Abstände der Buchstaben innerhalb eines Wortes dürfen beim *Randausgleich* des Blocksatzes keinesfalls verändert werden!

Als häufigste Kunstform beim Absatzlayout bedient man sich oft *zentriert* ausgerichteter Absätze (z.B. bei Titelseiten, Folien oder in Tabellen).

linker Spaltenrand	Und auch bei zentriert ausgerichteten Absätzen werden die Zeilen solange gefüllt, wie es die Spaltenbreite zuläßt. Anschließend werden die Zeilen unter Beibehaltung des natürlichen Wortabstandes auf die Spaltenmitte zentriert.	rechter Spaltenrand

Kerning und Kolumnen

Kerning und Kursivkorrektur. Werden alle Lettern im gleichen Abstand zu Worten aneinandergereit, entsteht ein sehr unruhiges Satzbild. Dies beruht auf den stark unterschiedlichen Formen unserer Buchstaben: einige (zum Beipiel „M") füllen ihre Box fast aus, andere (zum Beispiel „T") lassen viel leeren Raum. Buchstabenkombinationen wie „Te" erzeugen dann unkorrigiert eine Lücke im Graubild des Satzspiegels, die sofort unangenehm auffällt. Als Gegenmaßnahme werden solche Buchstaben enger zusammengerückt. Diesen Vorgang nennt man im elektronischen Satz „Kerning"[2] (Abb. 1.4 oben). Wechselt die Schriftart von geneigten zu aufrechten Schriften, entsteht das genau entgegengesetzte Problem: Die Worte stehen zu dicht zusammen. Das Einbringen zusätzlichen Zwischenraums wird *Kursivkorrektur* genannt (Abb. 1.4 unten).

Te   Te

*V*A   *V*A

Abbildung 1.4: Kerning und Italic-Korrektur: links ohne, rechts mit

Während LaTeX das Kerning selbsttätig durchführt, muß die Kursivkorrektur in bestimmten Fällen mit dem Befehl `\/` von Hand vorgenommen werden.

Ligaturen. Bestimmte Buchstabenkombinationen werden zu eigenen Lettern zusammengezogen. Der Grund ist wie beim Kerning eine Vereinheitlichung des Satz-Graubildes. Im deutschsprachigen Satz werden folgende Ligaturen verwendet:

ﬀ ﬁ ﬂ ﬃ ﬄ

Ligaturen werden von LaTeX automatisch gebildet. Da fremdsprachige Worte jedoch ohne diese Ligaturen gesetzt werden sollen, kann die Bildung von Ligaturen mit dem Kursivkorrekturbefehl `\/` verhindert werden.

Natürlicher Wortabstand. Zwischen den Worten einer Zeile wird beim deutschspachigen Textsatz zunächst ein Abstand von einem drittel „Geviert"[3] eingebracht (Abb. 1.5). Dieser Abstand wird auch der *natürliche Wortabstand* genannt, und man spricht dann von *Drittelsatz*. Auch LaTeX verwendet diesen Drittelsatz[4].

Das Blatt

(drittel) Geviert

Abbildung 1.5: Natürlicher Wortabstand

Kolumnen. Bei Erreichen einer vorgegebenen Länge werden die Zeilen vertikal untereinander zu einer *Kolumne* (Spalte) angeordnet. Die Eigenart der deutschen Sprache, relativ *lange* Worte zu bilden, erweist sich besonders beim Blocksatz dieser Kolumnen als Schwierigkeit. Damit hier der natürliche Wortabstand nicht zu sehr durch den Randausgleich verändert werden muß, sind Trennungen erforderlich. Da ein Übermaß getrennter Worte wiederum die Lesbarkeit eines Textes verringert, sollte die Kolumnenweite deutschsprachiger Dokumente mindestens das eineinhalbfache der Länge des Alphabets in Kleinbuchstaben des verwendeten Zeichensatzes betragen. Bei besonders schmalen Kolumnen ist gegebenenfalls der Flattersatz anzuwenden.

---

[2] Im Bleisatz werden hierzu spezielle Lettern, die sog. *Logotypen* gegossen, die jeweils eine solche Kombination enthalten. Ein Zusammenrücken einzelner Lettern ist wegen deren fester Dickte nicht möglich.

[3] Ein Geviert ist ein Quadrat, dessen Kantenlänge der Kegelstärke entspricht; bei einer 10pt Schrift beträgt die Kantenlänge des Gevierts also 10pt.

[4] Die französische und englische Typographie verwendet in der Regel einen Wortabstand von einem Halbgeviert. Im Drittelsatz wirkt die Zeile jedoch ruhiger und geschlossener.

Absatz- und Seitenumbruch

Die Absätze eines Textes werden solange vertikal untereinander angeordnet, bis das untere Ende des *Satzspiegels* (siehe unten) der Seite (oder der Spalte) erreicht ist. Der nachfolgende Absatz wird dann auf die folgende Seite (in die folgende Spalte) gesetzt. Diesen Vorgang nennt man *Seitenumbruch* (Spaltenumbruch).

Als Satzspiegel bezeichnet man die von den Seitenrändern eingefaßte Fläche, in welcher der Umbruch vorgenommen wird (Abb. 1.1.4).

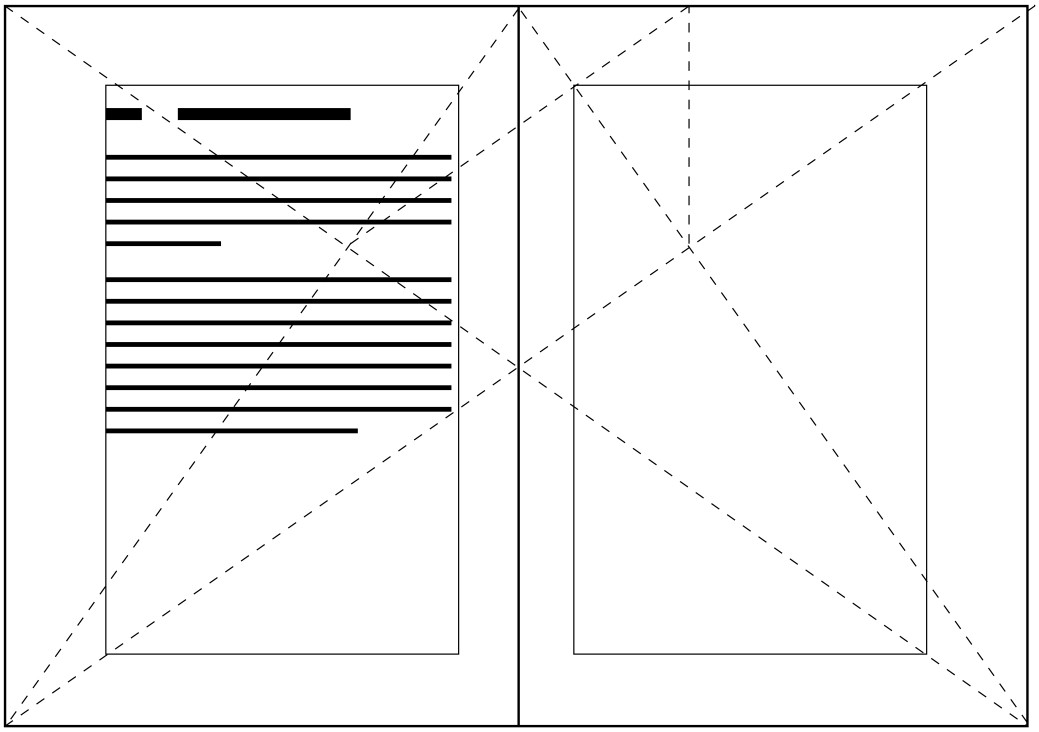


Abbildung 1.6: Satzspiegel nach dem goldenen Schnitt

Ist ein Absatz derart lang, daß beim Umbruch auf die folgende Seite (Spalte) die laufende Seite (Spalte) zu „leer“ würde, so wird der Absatz horizontal geteilt. Diesen Vorgang nennt man *Absatzumbruch*.

Beim Absatzumbruch ist unter anderem darauf zu achten, daß nicht nur eine einzelne Zeile auf die nächste Seite umgebrochen wird (sogenannte Schusterjungen).

1.1.5

Layout

Der Inhalt eines Textes wird beim Textsatz nur im Satzbild berücksichtigt (etwa bei Gedichten oder Formelsatz). Textkritisches Arbeiten ist kein Bestandteil des Textsatzes.

Satztechnisch baut sich ein Dokument aus Buchstaben verschiedener Zeichensätze, Worten, Zeilen, Absätzen, Figuren, Spalten und Seiten auf.

Den graphischen Anordnungsplan der Druckseite, also

- die Festsetzung des Satzspiegels und Anordnung der Spalten,
- die Wahl der Schriften für Fließtext, Überschriften etc.,
- die Ausführung/Anordnung von Fließtext, Überschriften...

nennt man das Layout des Textes. Im LaTeX-Programmsystem sind die Aufgaben wie folgt verteilt:

- LaTeX ist die Satz- und Layoutsprache, die die Rolle des „Layouters" für das Satzprogramm TeX übernimmt.

- LaTeX verfügt über eine Reihe von Standardlayouts, erlaubt jedoch auch die Gestaltung eigener Layouts mit sog. *Layoutbefehlen*.

- Layoutbefehle in LaTeX definieren die Funktionen der Elemente im Dokument. Diese in den Text eingefügten Befehle werden in Satzbefehle für TeX umgewandelt. Zum Beispiel bedeutet

  `\section{Layout}`: „Das ist eine Überschrift mit dem Inhalt Layout"

  `\begin{figure}`: „Jetzt fängt ein Bild an... ".

  `\end{figure}`: „...und hier hört es wieder auf."

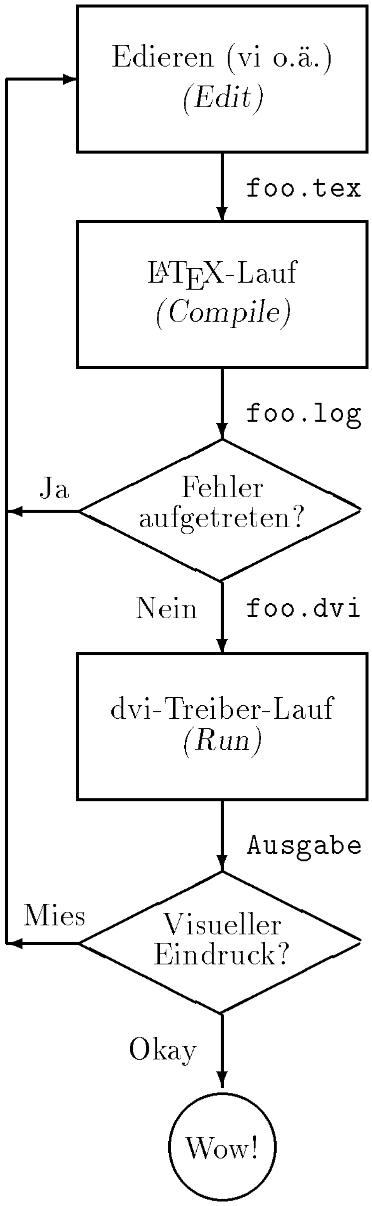
LaTeX ist im diesen Sinne eine „Markup-Language" wie z.B. HTML[5].

[5]**H**ypertext **M**arkup **L**anguage, populär z.B. im WWW.

1.1.6

Wie arbeitet man mit LaTeX?

Das folgende Beispiel arbeitet mit der Eingabedatei „`foo.tex`“. Der Dateiname kann frei gewählt werden, die Endung „`.tex`“ ist jedoch vorgesehen. Nach Wahl eines geeigneten Editors (z.B. `vi`, `emacs...`) erfolgt die Bearbeitung, ähnlich wie beim Programmieren in einer compilierenden Programmiersprache, in drei Schrittten (*Edit*, *Compile*, *Run*).



- Edieren einer LaTeX-Eingabedatei (erzeugen bzw. bearbeiten, *Edit*):

  ```
  emacs foo.tex
  ```

  Minimaler Inhalt:

  ```
  \documentclass{class}
  \begin{document}
       ⋮ (Hier steht der Text)
  \end{document}
  ```

  class = `article`, `report`, `book`
  oder andere Klassen

- LaTeX-Lauf (`foo.tex` mit LaTeX verarbeiten, *Compile*):

  ```
  latex foo.tex
  ```

- dvi-Treiber-Lauf (Kontrolle anhand eines Previews/Ausdrucks, *Run*):

  – Unter UNIX mit X11

  ```
  xdvi foo &
  dvips -P ps001_d4 foo
  ```

  – Unter DOS (emTeX)

  ```
  v foo
  prthplj foo
  ```

1.2 Struktur des LaTeX-Programmpakets

1.2.1

Die LaTeX-Eingabedatei

Die LaTeX-Eingabedatei ist eine normale ASCII-Datei, die mit jedem beliebigen ASCII-Editor erstellt und bearbeitet werden kann.

Den Inhalt einer solchen Eingabedatei analysiert LaTeX nach *Texteingaben* und *Befehlseingaben*. Texteingaben werden analog zu dem darin enthaltenen Text quasi *direkt* gesetzt, während Befehlseingaben auf die Art und Weise wirken, wie LaTeX den Text setzt, jedoch *nicht direkt* im Satz erscheinen.

- Außer Buchstaben und Zahlen akzeptiert LaTeX bei der Verarbeitung folgende Zeichen als Texteingaben:
  - Die Zeichensetzungszeichen: `. , : ; ! ? ‘ ’ -`
  - Runde und eckige Klammern: `( ) [ ]`
  - Einige Sonderzeichen: `+ = * / @`
  - Leerraum, wie z.B. Tabulator- oder Leerzeichen (Space)
- Die drei Sonderzeichen < > und | werden hauptsächlich im mathematischen Modus verwendet. Benutzt man sie als Texteingabe, so werden sie im Satz als andere Zeichen wiedergegeben.
- Die Sonderzeichen `# $ & ~ _ ^ \ { }` werden ausschließlich in LaTeX-Befehlen eingesetzt und haben in einer Texteingabe nichts verloren.
- Das Sonderzeichen % dient dazu, allen nachfolgenden Text in einer Zeile (inklusive des Zeilenendezeichens) zu überspringen (auszukommentieren).
- Ein guter Teil der oben erwähnten Sonderzeichen läßt sich durch simples Voranstellen des Rückwärtsschrägstrichs (Backslash) setzen:

```
\$ \& \% \# \_ \{ \} sind problemlos
```

$ & % # _ { } sind problemlos.

- Die deutschen Umlaute können durch Voranstellen von \" erzeugt werden:

schreibe	\"A	\"U	\"O	\"a	\"u	\"o	\ss
für	Ä	Ü	Ö	ä	ü	ö	ß

Für eine einfachere Umlautsyntax siehe Folie 1.2.5.

Eingabe von Text

Eingabe (Datei `foo.tex`):

```
Einfache Texte k\"onnen
f\"ur die Formatierung mit
\LaTeX\ eigentlich recht
einfach eingegeben werden.
Es kommt dabei nicht
darauf an,  wieviele
Leerzeichen     zwischen
  den    einzelnen
W\"ortern stehen, da
\LaTeX\ sie stets zu einem
einzigen Leerzeichen
zusammenzieht. Der
Zeilenumbruch und die
Trennungen erfolgen
unabh\"angig von der
Texteingabe automatisch.
Soll an einer bestimmten
Textstelle eine neue Zeile
beginnen, so mu\ss\ man das
explizit angeben.\newline
Die Umlaute und das
ess-zett (\ss) m\"ussen
gesondert behandelt werden.

Trifft \LaTeX\ bei der
Verarbeitung auf eine
Leerzeile in der Eingabe,
erzeugt es einen Absatz.



Mehrere Leerzeilen behandelt
\LaTeX\ (analog zu den
Leerzeichen) wie einzelne
Leerzeilen, es wird kein
zus\"atzlicher Platz
gelassen! \vspace*{\fill}
```

Ausgabe
(Datei `foo.dvi` durch dvi-Treiber):

Einfache Texte können für die Formatierung mit LaTeX eigentlich recht einfach eingegeben werden. Es kommt dabei nicht darauf an, wieviele Leerzeichen zwischen den einzelnen Wörtern stehen, da LaTeX sie stets zu einem einzigen Leerzeichen zusammenzieht. Der Zeilenumbruch und die Trennungen erfolgen unabhängig von der Texteingabe automatisch. Soll an einer bestimmten Textstelle eine neue Zeile beginnen, so muß man das explizit angeben.
Die Umlaute und das ess-zett (ß) müssen gesondert behandelt werden.

Trifft LaTeX bei der Verarbeitung auf eine Leerzeile in der Eingabe, erzeugt es einen Absatz.

Mehrere Leerzeilen behandelt LaTeX (analog zu den Leerzeichen) wie einzelne Leerzeilen, es wird kein zusätzlicher Platz gelassen!

Der LaTeX-Befehl

Die Textformatierung mit LaTeX wird durch *Befehle* in der Eingabedatei gesteuert.

- Befehle werden mit \ (Backslash) eingeleitet. Der Befehlsname folgt direkt danach.

- Ein Befehl wird durch ein Leer- oder Sonderzeichen abgeschlossen.

  Beispiele:

  ```
  Hier steht das \LaTeX -Symbol
  Hier steht das \LaTeX-Symbol
  ```

  Hier steht das LaTeX-Symbol

  ```
  Hier steht das \LaTeX\ -Symbol
  ```

  Hier steht das LaTeX -Symbol

  ```
  Hier steht ein \$ Zeichen
  ```

  Hier steht ein $ Zeichen

- Viele Befehle haben Argumente; solche in eckigen Klammern sind optional, Argumente in geschweiften Klammern müssen angegeben werden:

  `\befehl[`*opt1*,...,*optn*`]{`*arg1*`}...{`*argn*`}`

  Beispiel:

  `\documentclass[a4paper,german]{article}`

- Steht ein Befehl in einem durch geschweifte Klammern {...} eingeschlossenen Bereich (einer sog. *Gruppe*), so gilt er nur innerhalb dieser Gruppe.

  Beispiel:

  ```
  Eine Gruppe {\bfseries beschr"ankt} Hervorhebungen.
  ```

  Eine Gruppe **beschränkt** Hervorhebungen.

Auswahl eines Standardlayouts

Die erste Zeile einer LaTeX-Eingabedatei (z.B. `foo.tex`) legt fest, welcher *Dokumentklasse* das Dokument angehören soll:

`\documentclass{`*class*`}`

Eine Dokumentklasse definiert das Layout für alle Bereiche des Textes. Es stehen u.a. folgende Standardklassen zur Verfügung:

class	Beschreibung
article	Für kurze Schriftstücke mit einfacher Gliederung.
report	Für längere Schriftstücke; mehr Gliederungsebenen.
book	Für lange Schriftstücke, die aus mehreren Bänden bestehen können. Aufbereitung für doppelseitigen Druck.
slides	Für Folien.
proc	Ähnlich wie article, jedoch speziell für sog. Proceedings angepaßt.

Das Layout einer Dokumentklasse kann durch sog. *Optionen* beeinflußt werden; z.B. bereitet die Zeile

```
\documentclass[a4paper]{article}
```

den Eingabetext für das Papierformat Din A4 auf.

Es stehen u.a. Optionen für

- die Setzung des Papierformats,
- die Ausrichtung von Formeln,
- die Auswahl der Normalschriftgröße,
- die Aufbereitung gemäß doppelseitigem Satz

zur Verfügung.

Die Standardklassen

Eine Dokumentklasse ist nichts anderes als eine Sammlung von TEX- und LATEX-Befehlen, die vor dem eigentlichen Dokument des Benutzers abgearbeitet werden. Die Abarbeitung der Befehle wird durch die

`\documentclass...`

Zeile erreicht. Dies ist der Grund, warum diese Zeile am Anfang des Dokuments stehen muß. Die Klassenbefehle legen die Eigenschaften des Layouts für das gesamte Dokument fest und sind in Dateien mit der Endung `.cls` abgelegt (z.B. `report.cls`).

Bei der Verarbeitung einer LATEX-Quelle wird die Abarbeitung der Dokumentklasse im Logfile (`foo.log`) protokolliert. Die *report* Klasse meldet sich z.B. mit:

```
(/sw/share/ZDVtex-3.1415/texmf/tex/latex2e/base/report.cls
Document Class: report 1994/12/09 v1.2x Standard LaTeX document class
```

Derzeit werden die folgenden Klassen bei der Auslieferung von LATEX auf jeden Fall bereitgestellt [15]; sie werden daher *Standardklassen* genannt. Darüberhinaus kann es weitere Dokumentklassen geben, die jedoch nicht bei jedem LATEX-System installiert sein müssen:

article Die *article* Klasse, wie in [12] beschrieben.

report Die *report* Klasse, wie in [12] beschrieben.

book Die *book* Klasse, wie in [12] beschrieben.

slides Die *slides* Klasse, wie in [12] beschrieben. Diese Klasse entspricht dem Dokumentstil *slitex* wie er von LATEX2.09 bekannt ist.

ltxdoc Eine Klasse, mit der das LATEX Programm dokumentiert wird. Natürlich auch für andere Programme geeignet. Stilistisch lehnt sich diese Klasse an *article* an.

ltxguide Mit dieser Klasse werden z.B. LATEX Manuale geschrieben. Ein Beispiel ist [15]. Diese Klasse ist noch nicht ganz stabil; es sind weitere Änderungen angekündigt. Auch diese Klasse wurde von *article* abgeleitet.

ltnews Dokumentklasse für das *LATEX News information sheet*. Sinngemäß gilt das gleiche wie für *ltxguide*.

proc Diese Klasse ist besonders für den Satz von Proceedings geeignet. Auch diese Klasse wurde von *article* abgeleitet.

1.2.5 Anpassungen durch Pakete; deutsche Besonderheiten

Dokumentklassen legen das Layout für das ganze Dokument fest. Im Unterschied hierzu ändern sog. *Pakete* dieses vorausgewählte Layout oder fügen neue Formatierungsbefehle hinzu.

Ein solches Paket kann durch den Befehl

\usepackage[*option1, option2, ...*]{*pckg*}

geladen werden. Dieser Befehl muß in der *Präambel* stehen.

Beispiel:

```
\documentclass{article}
\usepackage[german]{babel}     } Präambel

\begin{document}
    ⋮
\end{document}
```

Das *babel*-Paket dient der vereinfachten Eingabe von Sonderzeichen und Umlauten und ersetzt Standardtexte für verschiedene, auswählbare Sprachen („Kapitel“ statt „chapter“ u.s.w.).

Mit der Option `german` stehen u.a. folgende Befehle zur Verfügung:

Für Umlaute, Anführungszeichen und „scharfes s“:

Schreibe	"A	"U	"O	"a	"u	"o	"s	"'	"`
statt	\"A	\"U	\"O	\"a	\"u	\"o	\ss		
für	Ä	Ü	Ö	ä	ü	ö	ß	„	“

Für Spezialfälle der deutschen Trennung:

Befehl	trennt	Beispiel	Ausgabe	
			ungetrennt	getrennt
"ck	ck als k-k	Dru"cker	Drucker	Druk-ker
"ll	ll als ll-l	Ro"lladen	Rolladen	Roll-laden
"|	Ligaturen	Auf"|lage	Auflage statt Auflage	Auf-lage

Standardpakete und Erweiterungen

Die Pakete bestehen ebenso wie die Dokumentklassen aus einer Sammlung von TEX- und LATEX-Befehlen. Im Unterschied zu den Klassen legen die Paketbefehle nur Teilbereiche des Layouts fest. Der vorwiegende Anwendungsbereich für Pakete liegt jedoch in der Erweiterung der Textsatzfähigkeiten der Standardklassen. Die Pakete stellen in diesem Sinn einen Baukasten dar, mit dem jeder Benutzer individuell die Eigenschaften zu LATEX hinzufügen kann, die er wirklich braucht.

Folgende Standardpakete werden derzeit ausgeliefert [15]:

doc Für die LATEX Dokumentation.

exscale Lädt skalierte Versionen der *math extension fonts*.

fontenc Stellt die Fontkodierung von LATEX ein.

ifthen Implementiert „if ... then ... else ...“ Komandos.

latexsym Lädt den *LATEX symbol font*.

makeidx Vereinfacht das Erstellen von Indexverzeichnissen.

newlfont Emuliert die alten LATEX2.09 Fontbefehle mit NFFS-Befehlen.

oldfont Emuliert die LATEX2.09 Fontbefehle.

syntonly Bearbeitet ein Dokument, ohne ein Dvi-File zu erzeugen.

tracefnt Regelt die Geschwätzigkeit der NFFS-Fontinfo.

Informationen über die Benutzung finden sich in den `.cls` oder `.dtx` Dateien des jeweiligen Pakets. Die meisten Pakete sind auch in [4] ausführlich beschrieben.

Es gibt weitere Sammlungen von Paketen. Eine der wichtigsten heißt *tool* und enthält u.a. die folgenden Pakete:

array Vereinfachter Tabellensatz.

dcolumn Implementiert Dezimaltabulator in Tabellen.

delarray Erzeugt Klammern um Matrizen.

tabularx Automatische Berechnung der Spaltenbreite in Tabellen.

afterpage Setzt Text auch außerhalb des Satzspiegels.

multicol Erlaubt bis zu 10-spaltigen Satz.

showkeys Druckt die verwendeten Label für Kreuzreferenzen aus.

fontsmpl Testet Fonts aus.

ftnright Druckt Fußnoten zweispaltig aus.

verbatim Verbesserte Version der *verbatim* Umgebung.

Trennungen

Normalerweise wird von LaTeX automatisch getrennt. Wenn Trennfehler auftreten, kann die Trennung eines Wortes mit dem Befehl \- explizit angegeben werden.

Beispiele: Nicht so

```
Es geht hier um die Texteingaben  als solche.
```

Es geht hier um die Tex-
teingaben als solche.

sondern so:

```
Es geht hier um die Text\-eingaben  als solche.
```

Es geht hier um die Text-
eingaben als solche.

Kommt ein schwer zu trennendes Wort mehrfach im Text vor, so ist es sinnvoll, dieses Wort in die Trennliste aufzunehmen. Dies geschieht mit dem Befehl \hyphenation{*pattern*}, der in der Präambel stehen muß. Dabei ist *pattern* ein Wort mit all seinen Trennmöglichkeiten. Die Trennstellen werden durch ein einfaches „-“-Zeichen angegeben (also *nicht* \-!).

Beispiel:

```
\hyphenation{Blumen-topf-erde
             Text-ein-gabe
             Oster-feuer}
```

Das *babel*-Paket, Akzente und Sonderzeichen

Akzente und Sonderzeichen. Bereits ohne Sprachanpassungen können die gebräuchlichen Akzente und Sonderzeichen der Sprachen mit lateinischen Alphabeten gesetzt werden. Dazu dienen die folgenden Befehle. Hierbei sind nicht alle möglichen Kombinationen von Buchstaben und diakritiscen Zeichen wiedergegeben worden; hier nicht aufgeführte Kombinationen lassen sich jedoch analog herstellen (z.B. `\c{c}` für ç).

Akzente

ö	`\"o`	ü	`\"u`
ß	`\ss`	ä	`\"a`
ō	`\=o`	ó	`\'o`
õ	`\~o`	ȯ	`\.o`
ǒ	`\v{o}`	o̧	`\c{o}`
ọ	`\d{o}`	o̱	`\b{o}`
ô	`\^o`	o͡o	`\t{oo}`
ŏ	`\u{o}`	ő	`\H{o}`

Sonderbuchstaben

ı	`\i`	ȷ	`\j`
ł	`\l`	å	`\aa`
œ	`\oe`	Ø	`\O`
ø	`\o`	æ	`\ae`
Å	`\AA`	Æ	`\AE`
Œ	`\OE`	Ł	`\L`
¿	`` ?` ``	¡	`` !` ``

Sonderzeichen

§	`\S`
†	`\dag`
©	`\copyright`
£	`\pounds`
¶	`\P`
‡	`\ddag`
\	`$\backslash$`

Das *babel*-Paket. Dieses Paket stellt Anpassungen für eine ganze Reihe von Sprachen bereit. Zum einen sind dies passende Trenntabellen (die jedoch bei der Installation des LaTeX-Programms explizit eingebunden werden müssen). Zum zweiten werden die in LaTeX vordefinierten Texte durch Begriffe der jeweiligen Landessprache ersetzt. So zum Beispiel „chapter" durch „Kapitel". Darüber hinaus werden in vielen Sprachen Befehle bereitgestellt, die die Eingabe der Sonderzeichen und diakritischen Zeichen vereinfachen. In vielen Fällen sind auch die nationalen typographischen Besonderheiten berücksichtigt.

Die Auswahl der unterstützten Sprachen erfolgt über Optionen; die Zeile

```
\usepackage[german,french]{babel}
```

lädt die Sprachunterstützung für Deutsch und Französisch. Innerhalb des Dokuments kann dann mit

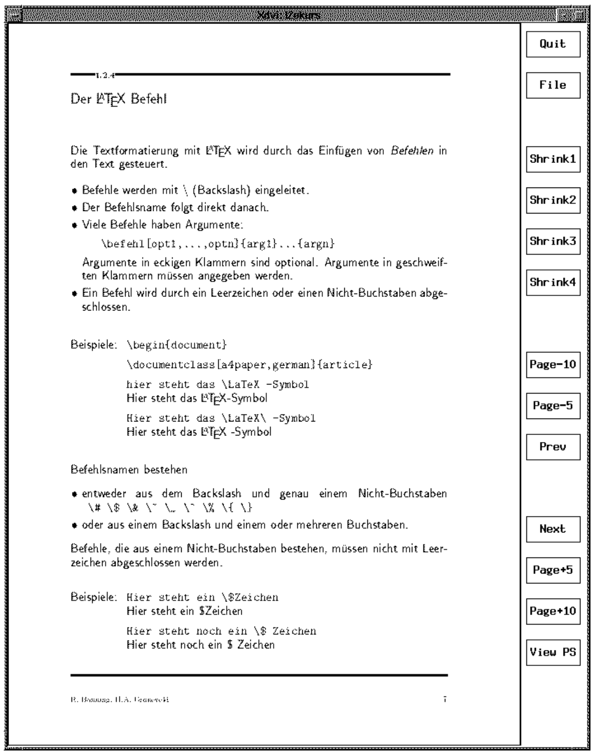
`\selectlanguage{`*lang*`}`

auf die Sprache *lang* umgeschaltet werden. Derzeit werden folgende Sprachen unterstützt:

lang	Option
Bahasa	`bahasa`
Dänisch	`danish`
Deutsch	`german`
Englisch	`english`
Esperanto	`esperant`
Finnisch	`finnish`
Französisch	`french`
Galizisch	`galician`
Italienisch	`italian`
Katalanisch	`catalan`
Kroatisch	`croatian`
Niederländisch	`dutch`
Norwegisch	`norsk`
Polnisch	`polish`
Portugisisch	`portuges`
Rumänisch	`romanian`
Schwedisch	`swedish`
Slowakisch	`slovak`
Slowenisch	`slovene`
Spanisch	`spanish`
Türkisch	`turkish`
Tschechisch	`czech`
Ungarisch	`magyar`

Visuelle Konrolle: die Previewer

Nach Be- und Verarbeitung einer LaTeX-Eingabedatei (`foo.tex`) sollte stets eine visuelle Kontrolle des Textsatzes, wie ihn die Ausgabedatei (`foo.dvi`) beschreibt, *vor* einem eventuellen Ausdruck bzw. einer Weiterbearbeitung erfolgen. Dazu stehen verschiedene Preview-Programme zur Verfügung:



- Das Programm xdvi kann auf allen X11-fähigen UNIX-Rechnern benutzt werden.
- Der Aufruf von xdvi für die Datei `foo.dvi` erfolgt unter UNIX mit

  ```
  xdvi foo &
  ```

  (Die Endung `.dvi` kann hier ausgelassen werden, sofern keine Datei des Namens `foo` existiert.)
- Dokumente können in verschiedenen Vergrößerungen angezeigt werden.
- Ist das Programm ghostscript installiert, so zeigt xdvi eingebundene PostScript-Dateien mit an.
- Weitere Informationen liefert

  ```
  man xdvi
  ```

- Auf DOS-Rechnern wird der Previewer (emTeX) mit

  ```
  v foo
  ```

  aufgerufen.

- Es können ebenfalls verschiedene Vergrößerungsstufen eingestellt werden. Ferner ist es möglich, mehrere Seiten auf einem Bildschirm anzeigen zu lassen, sowie ein eingebautes Lineal zu verwenden.

- Weitere Informationen enthält die Datei `\emtex\doc\dvidrv.dvi`.

Der Druckvorgang

Es gibt verschiedene Möglichkeiten, das Ausgabefile `foo.dvi` auszudrucken:

- Steht ein PostScript-fähiger Drucker zur Verfügung, läßt sich die dvi-Datei `foo.dvi` mit dem Befehl

  `dvips -o foo.ps foo`

  in die PostScript-Datei `foo.ps` umwandeln (die Endung `.dvi` muß nicht angegeben werden). Die Datei `foo.ps` kann dann ausgedruckt werden.

- Unter UNIX kann in der Regel mit dem Befehl

  `dvips -P` *druckername* `foo`

  der Druckvorgang der dvi-Datei `foo.dvi` direkt eingeleitet werden. Mit

  `dvips -P` *druckername* `-p` *anf* `-l` *end* `foo`

  werden nur die Seiten von Nummer *anf* bis Nummer *end* der dvi-Datei gedruckt. Weitere Informationen liefert der Befehl

  `man dvips`

  und finden sich in der dvi-Datei

  `/sw/share/ZDVtex-3.1415/texmf/doc/dvips/dvips.dvi`

- Für emTEX Benutzer unter DOS gibt es eine Reihe von Druckertreibern für die gebräuchlichsten Druckertypen. Der Aufruf erfolgt mit

  `prt`*name* `foo`

  (auch hier muß die Endung `.dvi` nicht angegeben werden). Hierbei können für *name* die Typkürzel `hplj`, `p6l` `p6m`, `p6h`, `FX` oder `ITO` eingesetzt werden. Mit dem Befehl

  `prt`*name* `/b` *anf* `/e` *end* `foo`

  werden nur die Seiten von Nummer *anf* bis Nummer *end* der dvi-Datei gedruckt. Weitere Informationen enthält die dvi-Datei

  `\emtex\doc\dvidrv.dvi`

TEX-Fehlermeldungen

Verschreibt man sich bei der Eingabe eines LATEX-Befehls (in `foo.tex`) oder verwendet falsche Befehle, so kann LATEX diese Datei nicht vollständig bearbeiten. Es kommt zu Programmunterbrechungen, die durch sogenannte Fehlermeldungen angezeigt werden. Da LATEX ein Makropaket ist, das aus TEX-Befehlen aufgebaut ist, folgen den eigentlichen LATEX-Fehlermeldungen oft weitere TEX-Fehlermeldungen, die bei der Abarbeitung und Expandierung eines fehlerhaften LATEX-Konstrukts entstehen. Fehlermeldungen sind eindeutig an einem vorangestellten Ausrufezeichen zu erkennen.

Die Logdatei. Alle Fehlermeldungen werden in die Logdatei (`foo.log`) geschrieben. Diese hat folgenden Aufbau:

1. Header: `This is TeX, Version ...`
Es folgen Informationen, welche Dokumentklasse und Pakete geladen werden.

2. LATEX-Fontinfo: Die ausgewählten Fonts werden angezeigt.

3. Das eigentliche Protokoll: Jede gesetzte Seite wird in eckigen Klammern angezeigt. Das Symbol `[5` bedeutet, daß mit dem Satz der Seite 5 begonnen wurde. Wird Seite 5 fehlerfrei gesetzt, so wird die eckige Klammer geschlossen: `[5]`. Tritt zuvor eine Fehlermeldung auf, so ist der Fehler auf Seite 5 oder früher zu suchen.

4. Eine kleine Statistik des Speicherbedarfs.

Besteht eine Quelldatei `foo.tex` aus mehreren Eingabedateien (z.B. `foo1.tex` und `foo2.tex`; siehe dazu S. 138), so wird das Einlesen der Quelle `foo1.tex` durch das Symbol `(foo1.tex` in der Logdatei gekennzeichnet. Ist diese Teildatei abgearbeitet, wird die runde Klammer geschlossen. Alle Ursachen für Fehlermeldungen, die innerhalb dieses runden Klammerpaars erscheinen, sind dann in der Datei `foo1.tex` zu suchen.

TEX-Fehlermeldungen. Die häufigste Fehlerusache in TEX/LATEX sind Tippfehler. Schreibt man zum Beispiel `\TEX` statt `\TeX` um das TEX-Symbol zu setzen, so meldet LATEX bei der Verarbeitung der Quelldatei (Compile):

```
! Undefined control sequence
l.5 Hier steht das \TEX
                        \ Symbol.
?
```

Die erste Zeile kennzeichnet die Fehlerart, die zweite Zeile gibt den Kontext an. Dabei bedeutet `l.5`, daß der Fehler in der 5. Zeile der aktuellen Quelldatei auftritt (s.o.). Das letzte Wort in der Kontextzeile vor dem Zeilenwechsel (hier:`\TEX`) bezeichnet die Stelle im Quelltext, an der der Fehler auftritt. Das Fragezeichen in der 4. Zeile fordert zu einer Aktion auf (Tasteneingabe); möglich sind u.a.:

`?` : Zeigt mögliche Aktionen an.
`r` : LATEX-Lauf ohne weitere Unterbrechungen fortsetzen.
`q` : Wie `r` mit nunmehr gänzlich unterdrückten Fehlermeldungen.
`e` : Editor aufrufen (Fehlerbeseitigung).
`h` : Zusätzlichen Hilfstext anzeigen.
`x` : LATEX-Lauf abbrechen.
<Return>: LATEX-Lauf fortsetzen.

Von allen Aktionen sei dem Benutzer vor allem das Drücken der <Return>-Taste empfohlen. Werden die Fehler in diesem Sinne zunächst ignoriert, lassen sie sich danach umso besser in Zusammenschau von Logdatei und Previewer lokalisieren, verstehen und beseitigen.

TEX- und LATEX-Fehlermeldungen

LATEX-Fehlermeldungen. Tritt bei der Abarbeitung von LATEX-Befehlen — das sind die in diesem Buch beschriebenen Befehle — ein Fehler auf, so meldet sich LATEX mit eigenen Fehlermeldungen. Diese unterscheiden sich formal von den TEX-Fehlermeldungen nur geringfügig durch einen Vorspann:

```
! LaTeX Error: Environment berbatim undefined.

See the LaTeX manual or LaTeX Companion for explanation.
Type  H <return>  for immediate help.
 ...

l.4 \begin{berbatim}
?
```

Hier wurde statt der *verbatim*-Umgebung (siehe S. 65) irrtümlich `\begin{berbatim}` aufgerufen. Für diese Fehlermeldungen gelten die gleichen Empfehlungen wie zuvor: zunächst mit <Return> versuchen, den LATEX-Lauf zuendezuführen, um dann mit Hilfe der Logdatei die aufgetretenen Fehler zu lokalisieren und zu verstehen.

Fehleranalyse. Gerade die Lokalisierung von Fehlern bereitet zuweilen Schwierigkeiten, da sich der Fehler *vor* der Stelle befinden kann, an der LATEX einen Fehler entdeckt. Dies ist z.B. dann der Fall, wenn vergessen wurde, eine Dokumentklasse mit `\documentclass` zu vereinbaren. Als Fehlermeldung erscheint dann die etwas geheimnisvolle Meldung:

```
! LaTeX Error: The font size
     command \normalsize is
     not defined:
     there is probably something
     wrong with the class file.
l.2 \begin{document}
```

Der Text in Zeile 2 ist natürlich zweifellos ein richtiges LATEX-Konstrukt. Der Fehler tritt als Konsequenz des falschen oder fehlenden Aufrufs in Zeile 1 (`\documentclass`) auf. Werden also Zeilen von TEX/LATEX als fehlerhaft bezeichnet, die unzweifelhaft syntaktisch richtig sind, so spricht man von einem Folgefehler, dessen Ursprung vor der bezeichneten Stelle liegt. Die Quelldatei (`foo.tex`) ist in diesem Fall „rückwärts“ zu lesen, bis man den eigentlichen Fehler findet. Ist dieser bei komplizierten Quelldateien nicht eindeutig lokalisierbar, hilft nur abschnittweises Auskommentieren der betroffenen Stellen mittels `%` (siehe S. 11 nebst anschließendem LATEX-Lauf. Tritt der Fehler nach einigen solcher Schritte nicht mehr auf, hat man den Fehlerort zumindest eingegrenzt.

Eine weitere Möglichkeit, Folgefehler zu erzeugen, birgt ein vergessener Umgebungsanfang. Vergißt man bei einer Aufzählung (siehe S. 72) die Zeile `\begin{itemize}`, so werden die `\item` Befehle nicht erkannt, da diese erst durch den Aufruf der Umgebung definiert werden. LATEX meldet in diesem Fall:

```
! LaTeX Error: Lonely
     \item--perhaps a missing
     list environment.
```

Liste der TEX-Fehlermeldungen. Eine vollständige Aufzählung der möglichen Fehler würde den Rahmen dieses Buches sprengen. Eine relativ vollständige und gut kommentierte Liste findet man aber z.B. in [10, Kapitel 9.].

Warnungen und Fehlermeldungen

Auch eine syntaktisch einwandfreie LaTeX-Quelle kann zu (typographisch) unbefriedigenden Ergebnissen führen. Dies wird von TeX/LaTeX weitgehend erkannt und durch Warnungen mitgeteilt, die während des Programmdurchlaufs (Compile) angezeigt werden.

TeX-Warnungen. Diese Warnungen beziehen sich ausschließlich auf die Güte des Zeilen- und Seitenumbruchs. Eine TeX-Warnung ist kein Fehler! Die Warnungen erscheinen beim LaTeX-Lauf (Compile) auf dem Bildschirm und werden ebenfalls in die Logdatei (`foo.log`) eingetragen. Die häufigste Warnung ist

```
Overfull \hbox (6.98082pt too wide)
   in paragraph at lines 18--21
   []\OT1/cmtt/m/n/14.4
   Hier steht das \LaTeX -Symbol
```

Das Schlüsselwort `Overfull` zeigt an, daß TeX im Absatz von Zeile 18 bis 21 eine Zeile nicht richtig umbrechen konnte und diese Zeile 6.98082pt (das sind gut 3mm) über den Rand hinausragt. Das Problem taucht an der Stelle „`Hier steht das \LaTeX -Symbol`" auf. Abhilfe kann das Einfügen von weiteren Trennhilfen (`\-`; siehe S. 18) sein. TeX beschwert sich bereits bei Überlängen von Bruchteilen eines pt. Eine Überlänge von weniger als 1pt (etwa 0.3 mm) kann als gerade noch akzeptabel angesehen werden. In einem solchen Fall ist die Warnung einfach zu ignorieren.

Zuweilen erscheint auch die Warnung `Overfull \vbox`. Damit weist LaTeX auf eine zu lange Seite hin. In aller Regel sind diese Warnungen auf zu hohe, nicht umbrechbare Boxen im Text — wie zum Beispiel Tabellen — zurückzuführen. In diesem Fall sollte die Box an eine andere Textstelle verlegt oder eine Floating Umgebung (siehe Seite 129 benutzt werden.

Den Fall „zu leerer" Textstellen zeigt die Warnung

```
Underfull \hbox (badness 10000)
   in paragraph at lines 86--92
   []\OT1/cmss/m/n/14.4 Mehrere
   Leer-zei-len be-han-delt
```

an. Hier empfindet TeX den Abstand der Worte zwischen Zeile 86 und 92 als zu groß. Die `badness` bewertet die Mangelhaftigkeit des erzielten Umbruchs. Dabei ist 10000 „unendlich schlecht", ein Wert bis 200 normal (keine Warnung) und ein Wert bis 2000 für den Benutzer zumeist tolerabel. Die Warnung wird oft durch unnötige oder unsinnige Zeilenunbruchbefehle `\\` hervorgerufen. Eingesetzte Trennhilfen (`\-`) können dagegen die `badness` verringern.

Gleiches gilt sinngemäß für die Warnung `Underfull \vbox`

LaTeX-Warnungen. Auch LaTeX kann Warnungen abgeben. Alle diese Warnungen beginnen mit `LaTeX Warning` oder `LaTeX Font Warning:`. Die erste Art der Warnungen bezieht sich in der Regel auf (noch) nicht aufgelöste Kreuzreferenzen (siehe Seite 92). Diese Warnungen sollten erst beachtet werden, wenn sie auch nach mehrfachen LaTeX-Läufen nicht verschwinden. Die zweite Art der Warnungen informiert den Benutzer in der Regel über das nicht Vorhandensein eines gewählten Fonts und den durch LaTeX stattdessen ausgewählten Ersatzfont.

1.3 Die Standarddokumente (article, report, book)

1.3.1

Übersicht

Die Standarddokumente unterscheiden sich weniger in der Formatierung des eigentlichen Textes, als in den begleitenden Textelementen wie Gliederungsüberschriften und Verzeichnissen.

Die folgende Tabelle zeigt die Voreinstellungen für die wichtigsten Dokumentklassen:

	book	report	article
Gliederungsebenen			
\ part	x		
\ chapter	x	x	
\ section	x	x	x
Inhaltsverzeichnis	bis subsection		bis **sub**subsection
Beidseitig	Standard	nur mit Dokumentklassenoption *twoside*	
Numerierung (table, figure)	Kapitelnummer + fortlaufend im Kapitel Bsp: 1.1, 1.2, 1.3... 2.1, 2.2,...		fortlaufend Bsp: 1, 2, 3,...
Abstract, Titelseite	eigene Seite		eigene Seite nur mit *titlepage* Option
Headings			
links:	chapter		section
rechts:	section		subsection

Im folgenden werden die Layoutelemente der Dokumentklassen im einzelnen behandelt.

1.3.2

Ein erster Blick auf Kopf- und Fußzeilen

Die Textelemente oberhalb und unterhalb des eigentlichen Textmaterials heißen *Kopf-* bzw. *Fußzeile*; es können darin zum Beispiel Überschriften und Seitenzahlen wiedergegeben werden:

4 *KAPITEL 1. Der Seitenstil*

Die Typographie befaßt sich mit der Gestaltung eines Textes durch Schriften und andere Zeichen. Wörtlich übersetzt bedeutet *Typographie* „Schreiben mit Typen“. Dies wird verständlich, wenn man bedenkt, daß im klassischen Buchdruck (Bleisatz) jeder Buchstabe spiegelverkehrt auf einem Stempel erhaben ausgebildet ist und *Type* oder auch *Letter* genannt

1.1. Kopf- und Fußzeilen 5

wird (Abb. 1.2). Aus den Lettern entstehen durch Aneinanderreihung Wörter. Durch die typographische Gestaltung soll der Leser bei der Aufnahme des Textinhalts unterstützt werden. Dies kann z.B. durch die geeignete Wahl der Schriftart, der Größe der Buchstaben und die Verteilung des Textes auf der Seite erreicht werden. Bei der Wahl der Ge-

Die Gestalt der Kopf- und Fußzeilen kann mit dem Befehl

`\pagestyle{`*Stil*`}`

festgelegt werden. Das obige Beispiel ist mit dem Stil `headings` erzeugt worden:

```
\documentclass[german,a4paper]{report}
\usepackage{babel}
\pagestyle{headings}
\begin{document}
\tableofcontents
\chapter{Der Seitenstil}
\section{Kopf- und Fu"szeilen}
Die Typographie befa"st sich
mit der Gestaltung eines Textes
durch Schriften und andere Zeichen ...
\end{document}
```

Es gibt verschiedene solcher Seitenstile für Kopf- und Fußzeilen, die auch die Art und Position der Seitennumerierung festlegen. Zum Beispiel erzeugt die Zeile

```
\pagestyle{empty}
```

Seiten ohne Kopf- und Fußzeilen sowie ohne Seitennumerierung.

Ein erster Blick auf Überschriften

> **Kapitel 1**
>
> **Textverarbeitung**
>
> **1.1 LaTeX2$_\varepsilon$ Eine neue ...**
>
> **1.1.1 Der augenblickliche Zustand: LaTeX 2.09**
>
> Das Textformatierungsprogramm LaTeX hat sich in den zurückliegenden Jahren ...
>
> **1.1.2 Warum LaTeX2$_\varepsilon$?**
>
> Schon heute haben diverse Erweiterungen dazu geführt, daß ...

Überschriften gliedern einen Text in Kapitel und Abschnitte.

Oft werden solche Abschnitte numeriert, wie dies links im Beispiel zu sehen ist.

Überschriften werden mit den Befehlen `\chapter` für die Kapitel, `\section` für die Hauptabschnitte usw. erzeugt.

```
\begin{document}
\chapter{Textverarbeitung}
\section{\ltwoe: Eine neue \LaTeX-Generation am \zdv}
\subsection{Der augenblickliche Zustand: \LaTeX~2.09}
Das Textformatierungsprogramm \LaTeX\ hat sich
in den zur"uckliegenden Jahren ...
\subsection{Warum \ltwoe?}
Schon heute haben diverse Erweiterungen dazu gef"uhrt,
da"s...

\end{document}
```

Die Schriftgröße und die Art der Numerierung wird von LaTeX automatisch gewählt, falls Gliederungsbefehle benutzt werden.

1.3.4

Ein erster Blick auf das Inhaltsverzeichnis

Inhaltsverzeichnis

Die mit den Gliederungsbefehlen erzeugten Überschriften können automatisch in einem Inhaltsverzeichnis zusammengefaßt werden. Dieses Inhaltsverzeichnis enthält auch die Seitenzahl, auf der der jeweilige Abschnitt beginnt.

Das Inhaltsverzeichnis wird von LaTeX angelegt, falls der Befehl

```
\tableofcontents
```

in der Quelldatei abgesetzt wird. Dies sieht z.B. wie folgt aus:

```
\documentclass[german,a4paper]{report}
\usepackage{babel}
\parskip1ex plus0.7ex minus0.3ex
\parindent 0em
\begin{document}
\tableofcontents
\chapter{Programmstruktur von \LaTeX}
\section{Was ist Textsatz?}
    ⋮
Hier steht der Haupttext
    ⋮
\end{document}
```

Das Inhaltsverzeichnis erscheint in der Dokumentklasse `report` vor dem Beginn des Textes auf einer eigenen Seite. Steht der Befehl `\tableofcontents` nach dem Haupttext, so setzt LaTeX das Inhaltsverzeichnis an den Schluß des Dokuments.

Bei Wahl der Dokumentklasse *article* erhält das Inhaltsverzeichnis nur mit der Klassenoption *titlepage* eine eigene Seite.

Seitenumbruch

Bei der Verarbeitung einer LaTeX-Quelldatei führt das Programm die Verteilung von Zeilen auf Absätze und von Absätzen auf Seiten automatisch durch. Diesen Vorgang nennt man Zeilen- bzw. Seitenumbruch.

- Der Seitenumbruch kann erzwungen werden mit:

  `\pagebreak[`*anz*`]`

  Beispiel:

```
...Zeichen.  W"ortlich "ubersetzt bedeutet
\emph{Typographie} "‘Schreiben mit Typen"’.
    \pagebreak
Dies wird verst"andlich, wenn man bedenkt,
da"s im klassischen Buchdruck (Bleisatz) ...
```

  ergibt:

8 *KAPITEL 1. Der Seitenstil*

Die Typographie befaßt sich mit der Gestaltung eines Textes durch Schriften und andere Zeichen. Wörtlich übersetzt bedeutet *Typographie* „Schreiben mit Typen“.

1.2. Seitenumbruch 9

Dies wird verständlich, wenn man bedenkt, daß im klassischen Buchdruck (Bleisatz) jeder Buchstabe spiegelverkehrt auf einem Stempel erhaben ausgebildet ist und *Type* oder auch *Letter* genannt wird (Abb. 1.2). Aus den Lettern entstehen durch Aneinanderreihung Wörter. Durch die typographische Gestaltung soll der Leser bei der Aufnahme des Textinhalts

- Seitenumbruch kann unterdrückt werden mit:

  `\nopagebreak[`*anz*`]`

  Dabei ist *anz* eine ganze Zahl zwischen 1 und 4.

- Ist *anz* kleiner als 4, so ist dies nur eine unverbindliche Empfehlung an LaTeX (mit wachsender Dringlichkeit). Bei *anz* gleich 4 gilt der Befehl absolut.

- Der Befehl `\newpage` bricht die laufende Seite mitten in der laufenden Zeile ab und beginnt eine neue Seite.

Zeilenumbruch

- Der Zeilenumbruch mit Randausgleich kann an jeder Stelle durch

  `\linebreak`

  oder

  `\linebreak[`*anz*`]`

  erzwungen oder erleichtert werden. Die Zahl *anz* kann zwischen 1 und 4 liegen. Ist *anz* gleich 4, wird ein unbedingter Zeilenumbruch durchgeführt, was gleichbedeutend mit dem Befehl `\linebreak` ist.

  Beispiel:

```
Die Typographie befa"st
sich mit der Gestaltung
eines Textes durch
Schriften und andere
Zeichen.  W"ortlich
eins zwei drei \linebreak
```

  ergibt:

  Die Typographie befaßt sich mit der Gestaltung eines Textes durch Schriften und andere Zeichen. Wörtlich
  eins zwei drei

- Analog kann ein Zeilenumbruch mittels

  `\nolinebreak`

  oder

  `\nolinebreak[`*anz*`]`

  an einer bestimmten Stelle unterdrückt werden.

- Der Befehl `\newline` oder `\\` bricht die laufende Zeile unmittelbar ab.

Abschnitt 2

Layout unter LaTeX

Die Verteilung des Textes auf der Seite, die Wahl der Zeichensätze, die Anordnung graphischer Elemente usw. — all das bestimmt den ersten Eindruck, den man von einem gedruckten Text gewinnt. Weit mehr noch als den ersten Eindruck, beeinflußt die Kombination solcher Gestaltungsmöglichkeiten die gute Erfaßbarkeit eines Textes.

Die Gestaltung eines Dokuments — das Layout — gliedert sich dabei in mehrere Ebenen. Umfassend legt das *Seitenlayout* die räumliche Verteilung und die Zeichensatzauswahl der Kolumnen, der Kopf- und Fußzeilen und der Überschriften fest. Zur Gestaltung eines längeren Dokuments gehören aber auch z.B. Fußnoten oder Randnotizen, mit denen sich ein Text übersichtlich kommentieren läßt. Auch helfen Inhalts- und Abbildungsverzeichnisse dem Leser bei der Orientierung im Text.

2.1 Gesamt- und Seitenlayout

2.1.1

Layout unter LaTeX

Es gibt in LaTeX vielfältige Möglichkeiten, das Layout eines Dokuments zu beeinflussen. Dabei ist stets wünschenswert, eine Layoutanpassung so vorzunehmen, daß sie mit dem *Gesamtlayout*, also dem Erscheinungsbild des gesamten Dokuments harmoniert.

Verschiedene Erweiterungsmechanismen in und um LaTeXstehen für solche „harmonischen“ Anpassungen zur Verfügung. LaTeX-Benutzer verspüren dennoch — vor allem beim abschließenden Layout ihrer Dokumente — oft den Wunsch nach gezielteren Layoutänderungen. Solche Änderungen können betreffen:

- Die Art und Weise, wie LaTeX Dokumentteile zählt.
- Die Abstände der Layoutelemente voneinander (Breiten).
- Die Auswahl der verwendeten Zeichensätze.
- Die Ausfürung bzw. Einbeziehung von Stilelementen wie z.B. Fußnoten.

Um LaTeX mitzuteilen, wie man das Layout zu ändern gedenkt, muß man in der Regel bestimmte LaTeX-*Register* manipulieren. Dies geschieht vermöge spezieller Befehle, die später auf Folie 2.1.5 erklärt werden.

Man unterscheidet zwei Arten von Registern in LaTeX:

- Zählregister (Zähler; enthalten ganze Zahlen)
- Längenregister (Abstände; enthalten eine Maßangabe/Länge)

Um Gebrauch von letzteren machen zu können, ist die Kenntnis der von LaTeX unterstützten Maßeinheiten vonnöten.

Maße

Folgende Maße[1] können in LaTeX verwendet werden:

- Feste Maße:

Einheit	
`mm`	Millimeter
`cm`	Zentimeter, 1 cm = 10 mm
`in`	Inch, 1 in $\approx$ 25 mm
`pt`	Point, 1 pt $\approx \frac{1}{3}$ mm
`pc`	Picas, 1 pc = 12 pt $\approx$ 4 mm

- Elastische Maße[2]:

Horizontal:

Einheit/Befehl	Bedeutung
`em`	Breite eines großen „M“ (im aktuellen Font)
`\enspace`	so breit wie eine Ziffer
`\quad`	so breit, wie ein Großbuchstabe hoch ist
`\qquad`	doppelt so breit wie `\quad`
`\fill`	so groß, wie der verbleibende Raum in der Zeile (theoretisch von 0mm bis ∞)

Vertikal:

Einheit/Befehl	Bedeutung
`ex`	Höhe eines kleinen „x“ (im aktuellen Font)
`\smallskip`	etwa $\frac{1}{4}$ Zeile
`\medskip`	etwa $\frac{1}{2}$ Zeile
`\bigskip`	etwa 1 Zeile
`\fill`	wie oben, jedoch bezüglich der Seitenlänge

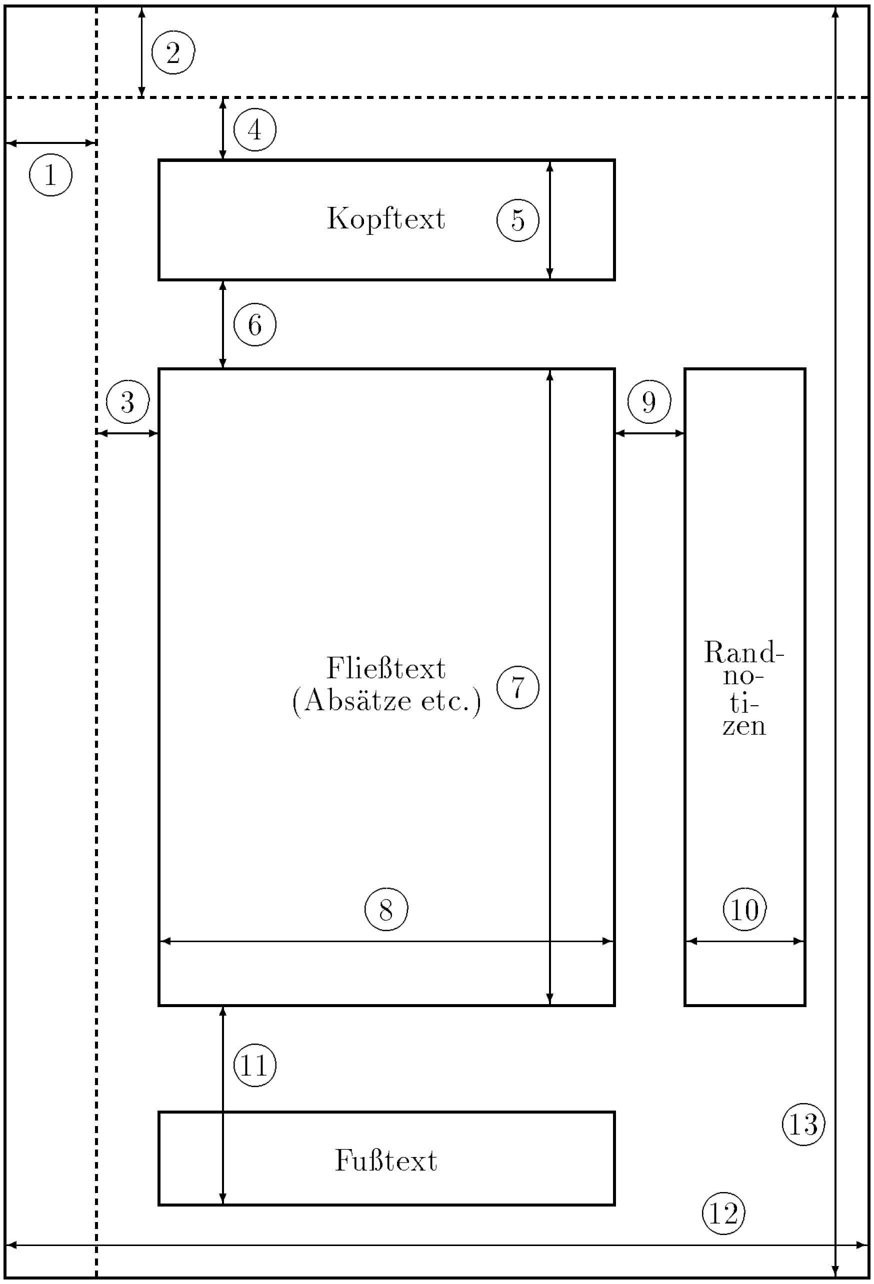
[1] Eine vollständige Maßangabe entsteht, indem man einem Maß eine — negative oder positive — Dezimalzahl voranstellt. Die Eins (pt = 1pt) sowie führende Nullen (.3em = 0.3em) können dabei weggelassen werden.

[2] Elastische Maße passen ihre Größe der jeweils gültigen Schriftgröße an. Sie sind daher besonders für zeichen- oder zeilenbezogene Abstände geeignet.

2.1.3

Dimensionierung des Layouts

In LaTeX ist das (einspaltige) Seitenlayout durch 13 Abstände definiert:



1
`1 in + \hoffset`

2
`1 in + \voffset`

3
`\odd`
`\evensidemargin`

4
`\topmargin`

5
`\headheight`

6
`\headsep`

7
`\textheight`

8
`\textwidth`

9
`\marginparsep`

10
`\marginparwidth`

11
`\footskip`

12
`\pagewidth`

13
`\pageheight`

Diese Abstände lassen sich in der Präambel mit dem Befehl
`\setlength{`*Abstand*`}{`*Maß*`}`
setzen. Weitere Befehle zur Längensetzung finden sich auf Folie 2.1.5.

2.1.4

Die Abstände im Seitenlayout

Diese Abstände dürfen ausschließlich in der Präambel gesetzt werden.

Abstand	Beschreibung
`\hoffset` `\voffset`	Zum Justieren der Papierränder; praktisch bei manchen Druckern oder Papierformatwechsel.
`\oddsidemargin` `\evensidemargin`	Einrückung auf den ungeradzahligen bzw. geradzahligen Seiten.
`\topmargin` `\headheight` `\headsep`	Abstand vom oberen Seitenrand zur Kopfzeile. Höhe der Kopfzeile. Abstand von der Kopfzeile zur Fließtextspalte.
`\textheight` `\textwidth`	Definieren Höhe und Breite des Fließtexts.
`\marginparsep` `\marginparwidth`	Abstand zwischen Randnotizenspalte und Fließtextspalte. Breite der Randnotizenspalte.
`\footskip`	Entfernung vom unteren Rand der Fließtextspalte zum unteren Ende der Fußzeilenbox.
`\pagewidth` `\pageheight`	Geben Breite und Höhe des Papiers an.

Mit den Befehlen `\enlargethispage{`*len*`}` und `\enlargethispage*{`*len*`}` läßt sich die Höhe der Fließtextspalte *einer Seite* um die Länge *len* vergrößern[3].

Alternativ lassen sich die oben genannten Abstände auch durch Verwendung einer Dokumentklassenoption auf vordefinierte Werte setzen:

Option (DIN)	für Papiergröße	
a4paper	210×297	mm
a5paper	148×210	mm
b5paper	176×250	mm

Option (US)	für Papiergröße	
letterpaper	8.5×11	inch
legalpaper*	8.5×14	inch
executivepaper	7.25×10.5	inch

* voreingestellt

Beispiel:

```
\documentclass[a5paper]{report}
```

[3] Die ∗-Form des Befehls versucht dabei, den Text maximal zu stauchen.

Verändern von Registern

Zähler: (Im folgenden steht *zähler* für einen beliebigen Zählernamen.)

- Mit den in der Tabelle genannten Befehlen lassen sich Zähler verändern:

Befehl	Bedeutung
`\value{`*zähler*`}`	Stellt den Wert eines Zählers bereit.
`\setcounter{`*zähler*`}{`*num*`}`	Setzt den Zähler auf den Wert *num*.
`\addtocounter{`*zähler*`}{`*num*`}`	Addiert den Wert *num* zum Zähler.
`\stepcounter{`*zähler*`}`	Erhöht den Zählerwert um 1.
`\refstepcounter{`*zähler*`}`	Wie `\stepcounter`, jedoch kann nun zusätzlich auf den Zähler mit `\ref` referenziert werden.

- Ein neuer Zähler cnt kann mit dem folgenden Befehl vereinbart werden:

  `\newcounter{cnt}[`*reset*`]`

  Der Zähler wird bei jedem Aufruf des Befehls *reset* auf Null zurückgesetzt.

Abstände: (Im folgenden steht *abstand* für einen bel. Abstandsnamen.)

- Abstände lassen sich mittels folgender Befehle verändern:

Befehl	Bedeutung
`\setlength{`*abstand*`}{`*len*`}`	Setzt den Abstand *abstand* auf *len*.
`\addtolength{`*abstand*`}{`*len*`}`	Erhöht den Abstand *abstand* um *len*.
`\settowidth{`*abstand*`}{`*text*`}`	Setzt den Abstand *abstand* auf die Breite des Textes *text*.
`\settoheight{`*abstand*`}{`*text*`}`	Setzt den Abstand *abstand* auf die Höhe des Textes *text* oberhalb der Grundlinie.
`\settodepth{`*abstand*`}{`*text*`}`	Setzt den Abstand *abstand* auf die Tiefe des Textes *text* unterhalb der Grundlinie.

- Ein neuer Abstand \lngth kann folgendermaßen vereinbart werden:

  `\newlength{\lngth}`

  Abstände werden bei ihrer Vereinbarung mit der Länge Null initialisiert.

Mehr zu Registern

- Der Wert eines jeden Zählers kann mit dem jeweils zu ihm gehörigen Befehl

  `\the`*zähler*

  ausgegeben werden.
- Es gibt eine Reihe weiterer Ausgabebefehle, die einen Zählerwert auf folgende Arten als Text setzen können:

  `\arabic{`*zähler*`}`
  Arabische Ziffern

  `\roman{`*zähler*`}`
  Römische Ziffern in Kleinbuchstaben

  `\Roman{`*zähler*`}`
  Römische Ziffern in Großbuchstaben

  `\alph{`*zähler*`}`
  Kleinbuchstaben, der Wert von *zähler* muß kleiner als 27 sein

  `\Alph{`*zähler*`}`
  Großbuchstaben, der Wert von *zähler* muß kleiner als 27 sein

Anmerkung: Während Zählregister einfache Integervariable sind, haben die dimensionsbehafteten Abstandsregister eher den Charakter eines Befehls (in der Syntax erkennbar am Backslash) und werden mitunter auch wie Befehle eingesetzt.

Beispiele:

- Um den Kapitelzähler auf Null zu setzen, kann man den Befehl

  ```
  \setcounter{chapter}{0}
  ```

  verwenden.
- Will man den Abschnittszähler auf den Wert des Kapitelzählers setzen, so geht das mit dem Befehl

  ```
  \setcounter{section}
            {\value{chapter}}
  ```
- Möchte man einen eigenen Zähler verwenden, der am Anfang jedes Abschnitts auf Null zurückgesetzt wird, so läßt sich das mit folgender Definition erreichen:

  ```
  \newcounter{mycount}[\section]
  ```

Beeinflussung des Layouts

Absätze:

Befehl	Bedeutung
`\sloppy`	Läßt LaTeX beim Trennen bzw. beim Austreiben und Einbringen „schlampiger“ vorgehen.
`\fussy`	Schaltet das Trennen wieder auf „pingelig“.
`\frenchspacing`	Unterdrückt den zusätzlichen Zwischenraum nach Satzzeichen.
`\parindent` †	Horizontaler Abstand, der die Einrückung der ersten Zeile eines Absatzes festlegt.
`\parskip` †	Zusätzlicher vertikaler Abstand vor Beginn eines neuen Absatzes; sollte eine dehnbare Länge sein.
`\baselineskip` ‡	Zeilenabstand innerhalb eines Absatzes.
`\baselinestretch` †	Dezimaler Faktor, mit dem im gesamten Dokument `\baselineskip` gewichtet wird. So kann man z.B. Text mit „anderthalbzeiligem“ Zeilenabstand setzen.

† Diesen Befehlen ist ein Maß zuzuordnen, z.B. mit
`\setlength{\parindent}{0mm} \setlength{\parskip}{.5em plus .5em minus .2em}`
`\renewcommand{\baselinestretch}{1.1}`
‡ Dieses Maß sollte nicht verändert werden.

Optionen für Dokumentklassen:

Option	Bedeutung
landscape	Vertauscht Seitenlänge und Seitenbreite $\Rightarrow$ Vorbereitung für Satz im Querformat.
10pt,11pt,12pt	Stellt die Größe der Normalschrift ein.
twoside	Satz gemäß doppelseitigem Layout $\Rightarrow$ Seitenzahlen stehen z.B. abwechselnd links und rechts.
draft	Markiert die Zeilen am Rand, die die Spaltenbreite überschreiten.
titlepage	Setzt in jedem Fall eine eigene Titelseite.
openany	Beginnt ein Kapitel auf einer beliebigen Seite (Voreinstellung: Kapitel beginnen nur auf ungeraden Seiten).
twocolumn	Zweispaltiges Seitenlayout.

Layoutanpassungen durch Pakete

Es stehen verschiedene Pakete zur Verfügung, die das Layout beinflussen können. Pakete lassen sich mit dem Befehl `\usepackage` in der Präambel laden.

Papierformat

Alternative Anpassung durch Pakete:

- Das Paket *a4* paßt das Format für DIN A4 Papier an. Die Spaltenhöhe beträgt 53, 46 oder 42 Zeilen jeweils für die Normalschriftgrößen 10pt, 11pt oder 12pt. Die Spaltenbreite ist eher schmal.
- Paket *a4wide* ⇒ Größere Spaltenbreite
- Mit dem Paket *anysize* stehen die gleichen Seitenoptionen wie in der Dokumentklasse zur Verfügung, jedoch ist die Option *a4paper* voreingestellt.

Beispiele:

```
\usepackage{a4}
\usepackage[b5paper]{anysize}
```

Seitenorientierung

Die Seitenorientierung (Hoch- oder Querformat) kann in LaTeX nur am Dokumentanfang festgelegt werden und gilt dann für das ganze Dokument. Mit dem Paket *portland* kann die Seitenorientierung für jede Seite getrennt festgelegt werden.

Mit `\usepackage{portland}`

wird das Paket geladen. Nun kann mit

`\landscape`

ins Querformat umgeschaltet und mit

`\portrait`

ins Hochformat zurückgewechselt werden. LaTeX beginnt jeweils eine neue Seite.

Statt dieser Befehle können auch die folgenden Umgebungen verwendet werden:

```
\begin{portrait} ... \end{portrait}
\begin{landscape}... \end{landscape}
```

Mehrspaltiges Layout

LaTeX stellt lediglich die Möglichkeit für durchgängig zweispaltigen Satz zur Verfügung (nun, immerhin). Mit dem Paket *multicol* kann Text in bis zu 10 Spalten gesetzt werden, und das flexibel mischbar auf unterschiedlichen Seiten. Dies Paket verführt allerdings leicht zu Experimenten, die einem harmonischen Gesamtlayout schaden können. Das Paket *multicol* wird auf Folie 3.1.6 ausführlich behandelt.

- Der Spaltenabstand `\columnsep` muß vor Aufruf der Umgebung *multicols* definiert werden.
- Zwischen den Spalten kann man einen senkrechten Strich der Breite `\columnseprule` setzen lassen (auch vor Umgebungsaufruf zu definieren).
- Um zuzulassen, daß die ganz rechte Spalte nicht unbedingt unten mit den übrigen Spalten bündig abschließen muß, dient der Befehl `\raggedcolumns`. Ist dieser gegeben, läßt sich ferner durch Setzen des Zählers `unbalance` auf einen positiven Wert der „Überstand" der ersten Spalten gegenüber der letzten (ganz rechten) Spalte einstellen.
- Nicht alle Möglichkeiten von LaTeX funktionieren in der *multicols* Umgebung uneingeschränkt (Abbildungen).

Beispiel:

```
\usepackage{multicol}
  :
\setlength{\columnsep}{5mm}
\setlength{\columnseprule}{0.1mm}
\begin{multicols}{2}[\section{Bla}]
\raggedcolumns
\setcounter{unbalance}{5}
```

Der Seitenstil

Der Seitenstil bestimmt das Satz- bzw. Erscheinungsbild von Kopf- und Fußzeile sowie das Erscheinungsbild der Seitenzahlen.

- Der Seitenstil kann in der *Präambel* gesetzt werden:

  `\pagestyle{`*style*`}`

- Außerdem läßt er sich für jede Seite individuell setzen:

  `\thispagestyle{`*style*`}`

- Als zwingende Argumente sind möglich:

style	Bedeutung
`plain`	Nur zentrierte Seitenzahl in der Fußzeile.
`empty`	Leere Kopf- und Fußzeile, keine Seitenzahlen.
`headings`	Kopfzeile mit Seitenzahl und laufender Überschrift, abhängig von der Dokumentklasse.
`myheadings`	Der Seitenkopf kann durch `\markright` bzw. `\markboth` selbst festgelegt werden.

- Der Seitenstil `headings` bestimmt den Seitenkopf automatisch.

- Bei `myheadings` wird der Seitenkopf durch

  `\markright{`*rechter Kopf*`}`

  bestimmt; ist doppelseitiges Layout aktiv (Option *twoside*) durch

  `\markboth{`*linker Kopf*`}{`*rechter Kopf*`}`

- Das Erscheinungsbild der Seitenzahlen kann mit dem Befehl

  `\pagenumbering{`*pstyle*`}`

  geändert werden.

- Mögliche Nummernstile *pstyle* sind:

pstyle	Bedeutung
Alph	große lateinische Buchstaben: A, B, C, ...
alph	kleine lateinische Buchstaben: a, b, c, ...
Roman	große römische Zahlen: I, II, III, IV, ...
roman	kleine römische Zahlen: i, ii, iii, iv, ...
arabic	arabische Ziffern: 1, 2, 3, ...

2.2 Fontauswahl

2.2.1

Definition eigener Befehle

Außer den von LaTeX vorgegebenen Befehlen lassen sich auch eigene Befehle definieren. Dies können z.B. selbstdefinierte Gliederungsbefehle oder Abkürzungen sein.

Ein neuer Befehl läßt sich wie folgt vereinbaren:

`\newcommand{`*cmd*`}[`#`][`*opt*`]{`*def*`}`

- *cmd* ist der neue Befehlsname, der allerdings noch nicht existent sein darf.
- *def* ist die Befehlsfolge, die bei Aufruf von *cmd* abgearbeitet werden soll.

Beispiel:

```
\newcommand{\bem}{\textbf{Bemerkung:}~}
\bem Dies ist ein selbstdefinierter Befehl.
```

Bemerkung: Dies ist ein selbstdefinierter Befehl.

- Selbstdefinierte Befehle können auch Argumente besitzen. Die Anzahl der Argumente wird optional durch die Zahl # definiert.
- Auf diese Argumente kann in *def* mit den Symbolen #1, #2, ...auf das erste, zweite, ... Argument zugegriffen werden.
- *opt* liefert den Default-Wert eines optionalen Arguments. Ein solches wird nur dann (unter dem Symbol #1) eingerichtet, wenn *opt* angegeben ist.

Beispiel:

```
\newcommand{\anvec}[2]{%
      \mbox{ $( {#1}_1,\ldots,{#1}_{#2} )$ }}
\anvec{x}{n}
```

$(x_1, \ldots, x_n)$

- Soll ein in LaTeX bereits existenter Befehl verändert werden, so ist hierzu der Befehl `\renewcommand` zu verwenden. Dieser hat die gleiche Syntax wie `\newcommand`.

Schriften

In der Typographie werden drei Merkmale einer Schrift beziehungsweise Schriftart unterschieden:

- Die *Schriftfamilie*,
- der *Schriftschnitt* und
- die *Auszeichnungsvariante*.

Die drei Merkmale kurz im einzelnen (Details auf den folgenden Seiten):

- Eine *Schriftfamilie* umfaßt solche Schriften, denen eine gemeinsame Gestaltungsidee zugrunde liegt, zum Beispiel:
  - Antiquaschriften (im folgenden englisch: Roman)
  - Sans Serif Schriften (wie auf diesen Folien verwendet)
  - `Schreibmaschinenschrift (englisch: typewriter)`
- Die *Schriftschnitte* liefern unterschiedlich *fette* Lettern einer Schriftfamilie. Im folgenden Beispiel sind drei Schriftschnitte der Schriftfamilie Sans Serif gezeigt.
  - Buchschrift
  - **Halbfetter Schriftschnitt**
  - **Fetter Schriftschnitt**
- Die *Auszeichungsvarianten* einer Schrift entstehen durch eine charakteristische Modifikation derselben, z.B. durch „Kippen“ der Lettern bei schräger oder auch bei kursiver Schrift. Die drei folgenden Beispiele diesmal auf Basis der Schriftfamilie Roman:
  - *Kursivschrift (englisch: italics)*
  - *Schrägschrift (englisch: slanted)*
  - Kapitälchen (englisch: small caps)

Mehr über Schriften

... über Schriftfamilien

- Roman Schriften (Antiqua) weisen charakteristische *Serifen* an den Buchstabenabschlüssen auf (kleine, wie mit einer Feder gesetzte Querstriche oder knotenartige Ausläufer). Roman Schriften zeichnen sich allgemein durch gute Lesbarkeit auch auf langen Zeilen aus.

- Sans Serif Schriften besitzen keine Serifen (daher der Name). Sie wirken nüchtern und klar, sind in schmalspaltigem Satz und in sehr großen Lettern (wie etwa auf Postern) ausgesprochen gut, jedoch auf langen Zeilen mitunter schwer lesbar. Sans Serif Schriften werden häufig beim Satz von (technischen) Handbüchern eingesetzt.

- Schreibmaschinenschriften sind (im Gegensatz zu Roman oder Sans Serif) keine *Proportionalschriften*. In einer Proportionalschrift ist ein „m" in der Regel breiter als ein „i", in einer Schreibmaschinenschrift hingegen haben alle Buchstaben die gleiche Breite. Da Terminalschriften in der EDV ebenfalls mit Buchstaben gleicher Breite arbeiten, nutzt man Schreibmaschinenschriften im Textsatz beispielsweise zur Dokumentation von Computerprogrammen.

... über Schriftschnitte

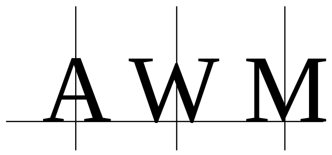
Meist sind Schriften einer Schriftfamilie in mehreren Schriftschnitten verfügbar, allerdings verwöhnen LaTeX bzw. TeX in dieser Hinsicht absichtlich nicht besonders, da unerfahrene Benutzer dazu neigen, verspielte Layouts mit vielen Schriften zu erzeugen, welche der Lesbarkeit eines Dokuments eher schaden. Erfahrene Benutzer werden, falls wirklich nötig, herausfinden können, wie man Fremdschriften einbindet oder selbst Schriften entwirft.

Eine Schriftfamilie behält in allen Schriftschnitten ihren Charakter bei, nur die Strichstärke variiert. Roman Schriften z.B. behalten ihre Serifen, Sans Serif Schriften bleiben ohne.

- Roman: Buchschrift, **Fetter Schnitt**
- Sans Serif: Buchschrift, **Halbfett**, **Fett**

... über Auszeichnungsvarianten

Standardschriften sind stets „aufrechte" Schriften, das heißt, die Symmetrieachse eines Buchstaben wie A, W oder M steht senkrecht auf der Grundlinie:



Bei den Auszeichungsvarianten der Schriften einer Familie muß dies nicht zutreffen. Wie der Name schon sagt, werden die Auszeichungsvarianten oft zur Hervorhebung (Auszeichnung) eines Wortes oder Textabschnitts verwendet.

- *Kursivschrift* (engl: italics) erinnert an Handschrift. Besonders die Kleinbuchstaben sind stark gerundet.
- *Schrägschrift* (engl: slanted) ist aus aufrechter Schrift durch Kippen der Symmetrieachse entstanden.
- KAPITÄLCHEN (engl: small caps) haben statt der Kleinbuchstaben kleine, fetter geschnittene Großbuchstaben.

Schriftfamilie

Eine Schriftfamilie umfaßt Schriften mit gleichen „Wesenszügen“, anders ausgedrückt, mit einer gemeinsamen Gestaltungsidee.

Name	Beispiel
Roman (Antiqua)	The quick brown fox jumps over the lazy dog
Sans Serif	The quick brown fox jumps over the lazy dog
Schreibmaschine	`The quick brown fox jumps over the lazy dog`

LaTeX kennt zwei Klassen von Befehlen, um die Schriftfamilie zu wechseln:

- Die erste Befehlsklasse wirkt als *Schalter*, das heißt, mit Aufruf eines der folgenden Befehle wird auf die jeweilige Schriftfamilie „umgeschaltet“. Um wieder die ursprüngliche Schriftfamilie zu benutzen, muß man wissen, welche das war, und auf diese mit dem entsprechenden Befehl „zurückschalten“.

  `\rmfamily` zum Einschalten der Schriftfamilie Roman.
  (Dies ist in allen Dokumentklassen voreingestellt.)

  `\sffamily` zum Einschalten der Familie Sans Serif.

  `\ttfamily` zum Einschalten von Schreibmaschinenschrift.

- Um kurze Textstücke in einer speziellen Schriftfamilie zu setzen, gibt es die zweite Klasse der Befehle zur Schriftfamilienwahl:

  `\textrm{...}` setzt den eingeklammerten Text in Roman,
  `\textsf{...}` in Sans Serif und
  `\texttt{...}` in Schreibmaschinenschrift.

Beispiel:

Folgende Befehlsaufrufe bewirken also das gleiche:

```
Es gibt \ttfamily verschiedene \sffamily Schriftarten.
```

```
Es gibt \texttt{verschiedene} Schriftarten.
```

Es gibt `verschiedene` Schriftarten.

2.2.4

Schriftschnitt

LaTeX stellt die Schriften der vorgestellten Schriftfamilien in verschiedenen *Schriftschnitten* zur Verfügung. Einfache Befehle zur Wahl des Schriftschnittes gibt es in LaTeX derzeit allerdings nur für Buchschrift und fetten Schnitt. Analog zur Auswahl einer Schriftfamilie gibt es auch hier zwei Befehlsklassen:

- Die erste Befehlsklasse wirkt als Schalter.

  `\mdseries` schaltet die Buchschrift ein.
  (Dies ist bei allen Dokumentklassen voreingestellt.)

  `\bfseries` schaltet einen fetten Schnitt ein.

- Für kurze Textstücke existiert die zweite Befehlsklasse.

  `\textmd{...}` wählt für den eingeklammerten Text die Buchschrift,
  `\textbf{...}` wählt einen fetten Schnitt der aktiven Familie.

Beispiele:

Die Wahl des Schriftschnittes ist mit der Wahl der Familie frei kombinierbar:

```
\rmfamily\bfseries Typographie
```

Typographie

```
\rmfamily\textbf{Typographie}
```

Typographie

```
\sffamily\bfseries Typographie
```

Typographie

```
\sffamily\bfseries Typographie \textmd{und} Druck
```

Typographie und **Druck**

Auszeichnungsvarianten

LaTeX kennt verschiedene Schriften zur Hervorhebung (Auszeichnung) eines Wortes oder Textabschnitts. Solche Schriften werden *Auszeichnungsschriften* oder *Auszeichnungsvarianten* einer Schriftfamilie genannt. Auch hier gibt es zwei Befehlsklassen zur Aktivierung der Schriften:

- Die erste Befehlsklasse wirkt als Schalter.

  `\itshape` schaltet die kursive Schrift (italics) ein.
  `\slshape` schaltet die Schrägschrift (slanted) ein.
  `\scshape` schaltet Kapitälchen (small caps) ein.
  `\em` schaltet die jeweilige Standardauszeichnungsvariante ein.
- Für kurze Textstücke gibt es die zweite Befehlsklasse.

  `\textit{...}` wählt kursive Schrift,
  `\textsl{...}` wählt Schrägschrift,
  `\textsc{...}` wählt Kapitälchen,
  `\textup{...}` wählt aufrechte Schrift,
  `\emph  {...}` wählt die Standardauszeichungsvariante.

Beispiele: (Es existieren nicht für jede Familie alle Auszeichnungsvarianten)

```
Roman: \textit{kursiv}, \textsl{schr"ag}
       und \textsc{Kapit"alchen}
```

Roman: *kursiv*, *schräg* und KAPITÄLCHEN

```
Sans Serif: \textsl{schr"ag}
```

Sans Serif: *schräg*

```
Schreibmaschine: \textit{kursiv}, \textsl{schr"ag}
                 und \textsc{Kapit"alchen}
```

Schreibmaschine: *kursiv*, *schräg* und KAPITÄLCHEN

```
Standardauszeichnung: \emph{kursiv} ...
```

Standardauszeichnung: *kursiv* (falls möglich)

Mehr zu Hervorhebungen

Zunächst sollte betont werden, daß oft eine elegante Hervorhebung bzw. Auszeichung durch Verwendung eines geeigneten Schriftschnitts anstelle einer Auszeichnungsvariante erreicht werden kann.

Auch die Wahl eines größeren (oder kleineren) Zeichensatzes ist gelegentlich der Benutzung von Auszeichungsvarianten vorzuziehen.

Doch um Teile eines Textes herzuheben, gibt es außer den rein typographischen auch noch verschiedene weitere, rein graphische Verfahren.

Für alle Hervorhebungsverfahren gilt, sie mit Bedacht einzusetzen, da eine allzu bunte Mischung verschiedenartiger Hervorhebungen einen Text unübersichtlich werden läßt. Auch Poster oder Folien profitieren in ihrer Lesbarkeit von einem eher sparsamen und vor allem systematischen Gebrauch typographischer wie graphischer Hervorhebungsverfahren.

- Das Unterstreichen von Textteilen ist eine allgemein bekannte und leider allzu weit verbreitete Möglichkeit. Sie sollte im Textsatz nicht oder nur ausnahmsweise benutzt werden.

  In LaTeX steht zum Unterstreichen der Befehl

  `\underline{`*text*`}`

  zur Verfügung. Dieser Befehl unterliegt jedoch zwei starken Einschränkungen, die zeigen, daß sein Einsatz in LaTeX wirklich nur für begründete Ausnahmefälle gedacht ist:

  Erstens werden unterstrichene Texte von LaTeX nicht umgebrochen, und einzelne unterstrichene Wörter deshalb nicht automatisch getrennt.

  Zweitens setzt LaTeX den Unterstreichungsstrich unterschiedlich tief, je nachdem, ob Unterlängen vorhanden sind, oder nicht (siehe das folgende Beispiel). Ein Unterstreichungsbefehl, der Unterlängen durchstreicht oder gar ausspart, existiert in LaTeX nicht, auch wenn er sich mit einiger Mühe zweifellos konstruieren ließe (mit Hilfe des `\rule`-Befehls nämlich).

Beispiel:

```
\underline%
{zusammenh"angend unterstrichen}
```

zusammenhängend unterstrichen

```
\underline{unzusammenh"angend}
\underline{unterstrichen}
```

unzusammenhängend unterstrichen

- Das g e s p e r r t e Setzen von Text ist eine nicht ganz so bekannte Hervorhebungsart wie das Unterstreichen, im Textsatz aber recht verbreitet und dem Unterstreichen allemal vorzuziehen. LaTeX bietet keinen eigenen Befehl für Sperrsatz an, daher muß man sich entweder mit eingeflochtenen `hspaces` oder mit „festen“ Leerzeichen — also Tilden (~) — behelfen.

Beispiel:

```
ein~~g~e~s~p~e~r~r~t~e~s~~Wort
```

ein g e s p e r r t e s Wort

Weitere Möglichkeiten, graphische Texthervorhebungen vorzunehmen, finden sich auf Folie 2.3.1.

Die Schriftgröße

Jede Schrift gibt es in verschiedenen Größen, die vom Benutzer mit folgenden Befehlen ausgewählt werden können (bezogen auf 10pt):

`\tiny`	5pt	Computer Modern Normal
`\scriptsize`	7pt	Computer Modern Normal
`\footnotesize`	8pt	Computer Modern Normal
`\small`	9pt	Computer Modern Normal
`\normalsize`	10pt	Computer Modern Normal
`\large`	12pt	Computer Modern Normal
`\Large`	14.4pt	Computer Modern Normal
`\LARGE`	17.28pt	Computer Modern Normal
`\huge`	20.74pt	Computer Modern Normal
`\Huge`	24.88 pt	Computer Modern Normal

Das Merkmal „Schriftgröße" legt zusammen mit den drei Merkmalen Schriftfamilie, Schriftschnitt und Auszeichnungsvariante eine Schrift eindeutig zu einem *Zeichensatz* (englisch: *Font*) fest.

- Die Schriften sind nicht linear skaliert, vielmehr besitzen kleine Schriften der gleichen Familie einen fetteren Schnitt:

blue	5pt Font auf 14.4pt skaliert
blue	14.4pt Font Entwurfsgröße

- Jeder Schriftgrößenbefehl gilt relativ zur gewählten Grundgröße von 10pt, 11pt oder 12pt.
- Jeder Schriftgrößenbefehl gilt, bis er durch einen anderen Schriftgrößenbefehl abgelöst wird.
- Soll ein Schriftgrößenbefehl in seiner Gültigkeit eingeschränkt werden, ist er in eine Gruppe zu klammern.

  Beispiel:

  ```
  Umschaltung in {\huge einer} Zeile.
  ```

  Umschaltung in einer Zeile.

Übersicht der Schriften

Es folgt eine Übersicht aller Schriften der Computer Modern Schriftfamilien. Diejenigen Schriften, die in der Spalte „Ansprechbar“ ein „ja“ eingetragen haben, lassen sich mit Hilfe der bisher erläuterten Befehle ansprechen.

Die anderen Schriften erfordern kompliziertere Befehle.

Roman			
Ansprechbar	Schnitt	Auszeichnung	Beispiel
ja	Buch	aufrecht	Computer Modern
ja	Buch	kursiv	*Computer Modern*
ja	Buch	schräg	*Computer Modern*
ja	Buch	Kaptitälchen	COMPUTER MODERN
ja	Fett	aufrecht	**Computer Modern**
ja	Fett	kursiv	***Computer Modern***
ja	Fett	schräg	***Computer Modern***

Sans serif			
Ansprechbar	Schnitt	Auszeichnung	Beispiel
ja	Buch	aufrecht	Computer Modern
ja	Buch	schräg	*Computer Modern*
ja	Fett	aufrecht	**Computer Modern**
nein	Halbfett	aufrecht	**Computer Modern**

Schreibmaschine			
Ansprechbar	Schnitt	Auszeichnung	Beispiel
ja	Buch	aufrecht	`Computer Modern`
ja	Buch	kursiv	*Computer Modern*
ja	Buch	schräg	*Computer Modern*
ja	Buch	Kaptitälchen	COMPUTER MODERN

Fibonacci			
Ansprechbar	Schnitt	Auszeichnung	Beispiel
nein	Buch	aufrecht	Computer Modern

Funny Roman			
Ansprechbar	Schnitt	Auszeichnung	Beispiel
nein	Buch	aufrecht	Computer Modern
nein	Buch	kursiv	*Computer Modern*

Dunhill			
Ansprechbar	Schnitt	Auszeichnung	Beispiel
nein	Buch	aufrecht	Computer Modern

2.3 Randnotizen, Überschriften und Fußnoten

2.3.1

Randnotizen

Randnotizen werden mit dem Befehl

\marginpar{*randnotiz*}

Diese
Box
ist
1,9cm
breit

erzeugt. Ihre Position relativ zum umgebenden Text liegt so, wie man es am Rand dieser Folie sieht.

```
...Ihre Position relativ zum
\marginpar{Diese\\Box\\ist\\1,9cm\\breit}
umgebenden Text liegt so,...
```

Der Text wird in einer Absatzbreite von ca. 1,9 cm gesetzt. Da das erste Wort eines Absatzes normalerweise nicht getrennt wird, kann es zu Problemen beim Umbruch kommen. Dies läßt sich umgehen, indem man dem ersten Wort den Befehl \hspace{0mm} voranstellt.

- Wichtige Textstellen können mit Hilfe eines Balkens hervorgehoben werden.

```
Wichtige \marginpar{\rule[-4ex]{1ex}{6ex}}
Textstellen k"onnen mit Hilfe eines Balkens ...
```

- Für doppelseitigen Textsatz gibt es eine erweiterte Syntax:

  \marginpar[*l_notiz*]{*r_notiz*}

$\Longrightarrow$ **Beispiel:**

```
\marginpar[ \hfill $\Longrightarrow$ ]{%
$\Longleftarrow$ }
```

- Normalerweise werden Randnotizen an den äußeren Rand gesetzt; bei einseitigem Textsatz ist dies der rechte Rand. Mit Aufruf des Befehls

  \reversemarginpar

  wird von da an der innere Rand für Randnotizen verwendet. Mit

  \normalmarginpar

  kann wieder auf Verwendung des äußeren Randes zurückgeschaltet werden.

2.3.2

Gliederungsbefehle

Für die Gliederung längerer Dokumente stellt LaTeX Gliederungsbefehle bereit, die unter anderem dazu dienen, die Überschriften in einem Dokument zu erzeugen und gegebenenfalls in ein Inhaltsverzeichnis einzutragen.

Beispiel:

```
\section{Einleitung}
\subsection{Erste Beispiele}
\subsubsection{Die logistische Gleichung}
```

1 Einleitung

1.1 Erste Beispiele

1.1.1 Die logistische Gleichung

Diese Befehle sind wie folgt aufgebaut:

gbefehl [*Kurzform*] {*Überschrift*}

oder

gbefehl∗{*Überschrift*}

- Bei der ersten Form wird die *Überschrift* in das Inhaltsverzeichnis übernommen und numeriert.
  Im optionalen Argument kann eine *Kurzform* der Überschrift angegeben werden, die besser in das Inhaltsverzeichnis paßt.
- Die ∗-Form von *gbefehl* erzeugt keine Numerierung der Überschrift und keinen Eintrag im Inhaltsverzeichnis.

Folgende Gliederungsbefehle *gbefehl* können verwendet werden:

gbefehl	Bedeutung
`\subparagraph`	5- bis 6-gliedrige Nummer.
`\paragraph`	4- bis 5- gliedrige Nummer.
`\subsubsection`	Unterunterabschnitt; 3- bis 4-gliedrige Nummer.
`\subsection`	Unterabschnitt; 2- bis 3-gliedrige Nummer.
`\section`	Abschnitt; 1- bis 2-gliedrige Nummer.
`\chapter`	Kapitel; nur in Dokumentklasse *book* oder *report*; einfache Numerierung mit zweizeiligem Aufbau.
`\part`	Band eines Buches; nur in Dokumentklasse *book*.

Die Gliederungszähler

Zu jedem Gliederungsbefehl gehört ein gleichnamiger Zähler
(also ohne den führenden Backslash „\“!).

Diese Zähler können mit den üblichen Befehlen verändert werden.
Dadurch ist es möglich, auch Teile von Dokumenten mit LaTeX zu bearbeiten.

Beispiel:

Ein Dokument soll mit dem dritten Kapitel und dem vierten Abschnitt beginnen:

```
\documentclass[11pt,german]{report}
 :
\begin{document}
 :
\setcounter{page}{140}
\setcounter{chapter}{2}

\chapter{Mathematische Formeln}
\setcounter{section}{3}
\section{Die {\tt array} -- Umgebung}

Zum Aufbau von Matrizen
und Gleichungsfeldern\ldots
 :
\end{document}
```

Kapitel 3

Mathematische Formeln

3.4 Die `array` – Umgebung

Zum Aufbau von Matrizen und Gleichungsfeldern...

Dies wurde also, wie man im Beispiel sieht,
durch eine entsprechende Setzung der Zähler
`\chapter` und `\section` erreicht.

Veränderung der Überschriften

Das vordefinierte Layout der Überschriften genügt nicht allen Anwendungen. Es gibt daher zwei Möglichkeiten, das Layout der Gliederungsbefehle zu ändern: `\@startsection` und `\secdef`.

- Mit `\@startsection` können alle Gliederungsbefehle von `\subparagraph` bis `\section` verändert werden. Dieser Befehl funktioniert nur innerhalb von Paket-Dateien (siehe Beispiel nächste Seite).
- Mit `\secdef` können *alle* Gliederungsbefehle verändert werden. Da dieser Befehl jedoch wesentlich bessere Kenntnisse der „LaTeX-Interna" erfordert, wird er nur auf den Ergänzungsseiten behandelt.

Der `\@startsection` Befehl kann nur in der Präambel abgesetzt werden. Der Befehl hat den folgenden Aufbau:

`\@startsection{`*name*`}{`*lev*`}{`*ind*`}{`*bef*`}{`*aft*`}{`*style*`}`

Die Bedeutungen der Befehlsparameter finden sich in der folgenden Tabelle:

Parameter	Bedeutung
name	Name des zu definierenden oder zu ändernden Gliederungsbefehls.
lev	Legt die Gliederungstiefe des Befehls fest und damit die Gliedrigkeit der Gliederungsnummer.
ind	Einrückung der Überschrift (Länge); bei negativer Einrückung beginnt die Überschrift außerhalb des linken Spaltenrandes.
bef	Vertikaler Abstand zum vorausgehenden Text; bei negativem Wert wird die Einrückung des nachfolgenden Absatzes unterdrückt.
aft	Vertikaler Abstand zwischen der Überschrift und dem nachfolgenden Absatz; bei negativem Wert wird die Überschrift in den Absatz eingefügt (eingezogene Überschrift), bei positivem Wert wird sie abgesetzt.
style	Schrift und Größe der Überschrift.

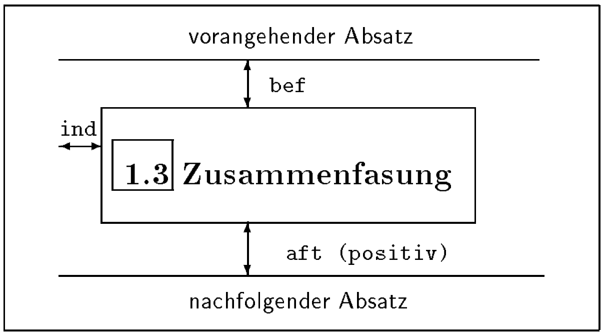
Überschriften mit `startsection`

Die folgenden Beispiele müssen in eine Datei mit der Endung `.sty` geschrieben werden, z.B. `mysection.sty`. Diese Datei ist dann mit

```
\usepackage{mysection}
```

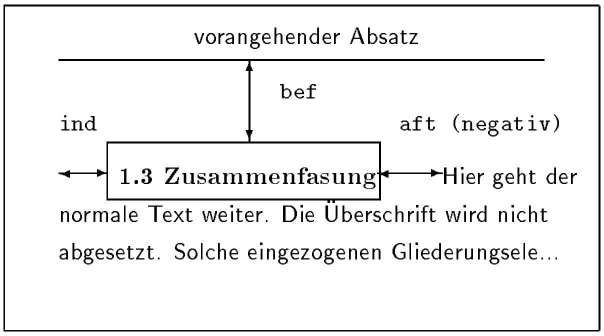
in der Präambel zu laden.

Beispiel 1: Abgesetzte Überschrift




```
\renewcommand{\subsection}{%
   \@startsection%
   {subsection}%
   {2}%
   {0mm}%
   {\baselineskip}%
   {.5\baselineskip}%
   {\rmfamily\bfseries}}
```

Beispiel 2: Eingezogene Überschrift




```
\renewcommand{\subsection}{%
   \@startsection%
   {subsection}%
   {2}%
   {0mm}%
   {\baselineskip}%
   {-.5\baselineskip}%
   {\rmfamily\bfseries}}
```

Noch ein Beispiel: Überschriften mit `secdef`

Die Änderung bestehender Gliederungsbefehle sei an einem Beispiel verdeutlicht. Die Wirkung der nachfolgenden Definition kann an den abgedruckten Folien der vorliegenden Kursunterlagen studiert werden, da selbige mit diesen Befehlen formatiert wurden.

Zunächst wird `\section` redefiniert:

```
\renewcommand{\section}
  {\secdef \Section \sSection}
```

Daraufhin wird der Befehl `\Section` definiert, der fortan bei Aufruf von `\section` abgearbeitet werden soll:

```
\newcommand{\Section}[2][?]{%
    \refstepcounter{section}%
    \addcontentsline{toc}{%
        section%
        }%
    {\protect\numberline{%
        \thesection } #1%
    }%
    {\LARGE\thesection~%
         \sffamily#2\par%
    }%
    \sectionmark{#1}%
    \vspace{\baselineskip}%
}%
```

1 notw.,1 opt. Arg., Default = ?
Zähler um 1 erhöhen
Eintrag ins TOC beginnt hier ...

... und endet hier.

Formatierung der Überschrift

Nummer erzeugen
zusätzlicher Abstand

Abschließend ist noch der Befehl `\sSection` zu definieren, den LaTeX aufgrund obiger Redefinition fortan immer dann benutzen wird, wenn man in der Eingabedatei den Befehl `section*` aufruft:

```
\newcommand{\sSection}[1]{%
    {\LARGE\sffamily#1\par}%
    \vspace{\baselineskip}%
}%
```

nur 1 notw. Argument
Formatierung der Überschrift
Abstand ... das war's

Fußnoten

Auf eine Fußnote wird üblicherweise durch ein hochgestelltes Symbol, meist eine Zahl[4], referenziert.

Der LaTeX-Befehl zur Erzeugung einer Fußnote lautet:

`\footnote{`*Fußnotentext*`}`

- Der Fußnotenbefehl kann nur im Fließtext verwendet werden.
- Die Numerierung ist bei der Dokumentklasse *article* fortlaufend für das ganze Dokument, bei `report` und `book` hingegen beginnt die Zählung mit jedem Kapitel neu. Zum Zählen wird der Fußnotenzähler `footnote` benutzt.
- Soll z.B. bei der Dokumentklasse *article* nach jedem `\section`-Befehl neu mit der Fußnotenzählung angefangen werden, so ist vor jedem `\section`-Befehl

  `\setcounter{footnote}{0}`

  einzugeben.
- Im Mathematischen Modus, in Tabellen, in Überschriften usw. kann der `\footnote` Befehl nicht verwendet werden. Um in solchen Fällen Fußnoten zu erzeugen, wird das Fußnotensymbol innerhalb der Formel, Tabelle, Überschrift... eingegeben, und der zugehörige Text außerhalb derselben:

  `\footnotemark[`*num*`]`
  `\footnotetext[`*num*`]{`*fußnotentext*`}`
- Jeder Aufruf von `\footnotemark` zählt den Fußnotenzähler weiter. Werden daher mehrere Marken vor dem ersten `\footnotetext` Befehl verwendet, muß der Fußnotenzähler von Hand auf den richtigen Wert zurückgesetzt werden.

[4]Man kann in LaTeX jedoch auch auf andere Symbole zurückgreifen.

Beispiele zu Fußnoten

Beispiel 1:

```
F"ur die Seiten eines rechtwinkligen Dreiecks gilt:
($a^2= b^2 + c^2$)\footnote{Dabei bezeichnet $a$ die Hypothenuse}
```

Für die Seiten eines rechtwinkligen Dreiecks gilt:
($a^2 = b^2 + c^2$)[1]

[1] Dabei bezeichnet a die Hypothenuse

Beispiel 2: (Macht das gleiche wie Beispiel 1, aber diesmal mit Hilfe von `\footnotemark` im mathematischen Modus und dann mit `\footnotetext` außerhalb desselben.)

```
F"ur die Seiten eines rechtwinkligen Dreiecks gilt:
$(a^2= b^2 + c^2)\footnotemark$
\footnotetext{Dabei bezeichnet $a$ die Hypothenuse}
```

Für die Seiten eines rechtwinkligen Dreiecks gilt:
$(a^2 = b^2 + c^2)^2$

[2] Dabei bezeichnet a die Hypothenuse

Beispiel 3: (Zurücksetzen des Fußnotenzählers.)

```
Wir haben also $[x,yz] + [y,zx] + [z,xy]\footnotemark$\\
und $[x,yz] = y[x,z] + [x,y]z\footnotemark$.
\addtocounter{footnote}{-1}\footnotetext{Jakobi-Identit"at}
\stepcounter{footnote}\footnotetext{Derivationseigenschaft}
```

Wir haben also $[x, yz] + [y, zx] + [z, xy]^3$
und $[x, yz] = y[x, z] + [x, y]z^4$.

[3] Jakobi-Identität
[4] Derivationseigenschaft

2.3.6

Ändern des Fußnotenstils

Verwendet man das optionale Argument des `\footnote`-Befehls, so wird das der Zahl num zugeordnete Fußnotensymbol verwendet, *ohne* daß sich der Zähler `footnote` verändert.

`\footnote[`num`]{`$fu\beta notentext$`}`

Soll der Fußnotenzähler `footnote` in einem anderen Stil ausgedruckt werden, oder anders gesagt, soll mit anderen Symbolen als mit arabischen Zahlen auf Fußnoten referenziert werden, so läßt sich dies mittels des Befehls

`\renewcommand{\thefootnote}{\`*ausgabestil*`{footnote}}`

erreichen. Als *ausgabestil* ist außer `\arabic`, `\roman`, `\Roman`, `\alph` und `\Alph` für `footnote` auch `\fnsymbol` möglich. Den Zählerwerten von 1 bis 9 entsprechen bei letztem die Symbole

$*$ † ‡ § ¶ ‖ $**$ †† ‡‡

Beispiel:[‡‡]

```
\renewcommand{\thefootnote}{\fnsymbol{footnote}}
\textbf{Beispiel:}\footnote[9]{Als
Markierung wurde Symbol 9 verwendet}
\renewcommand{\thefootnote}{\arabic{footnote}}
```

In der *minipage* Umgebung wird mit `mpfootnote` ein eigener Fußnotenzähler verwendet, der im Normalfall als Fußnotensymbol kleine lateinische Buchstaben verwendet.

Beispiel:

```
\begin{minipage}{10cm}
Diese Minipage enth"alt eine Fu"snote.\footnote{Diese
Fu"snote wird unmittelbar nach Ende der Minipage gesetzt.}
\end{minipage}
```

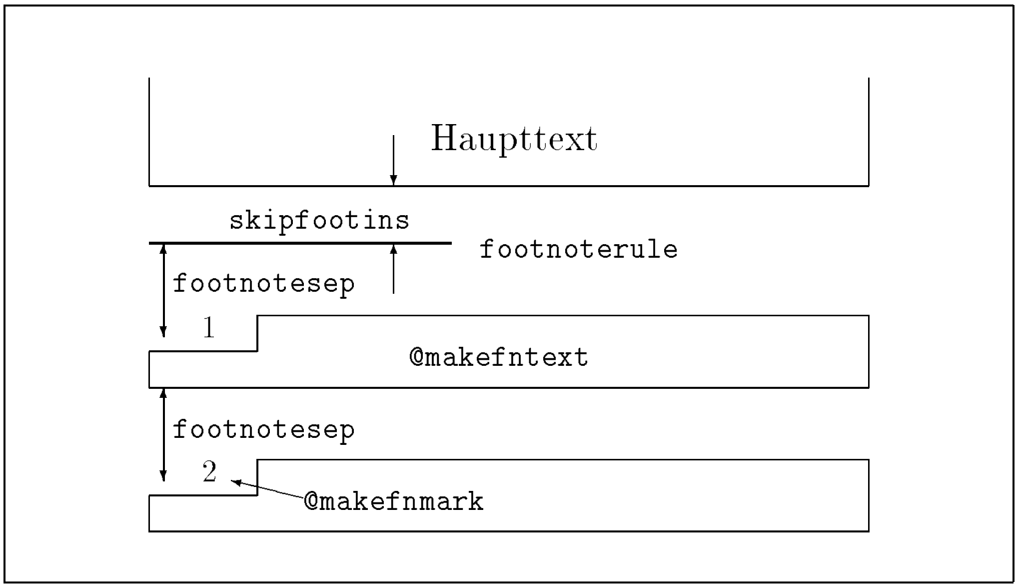
Diese Minipage enthält eine Fußnote.[a]

[a]Diese Fußnote wird unmittelbar nach Ende der Minipage gesetzt.

[‡‡]Als Markierung wurde Symbol 9 verwendet.

Ändern des Fußnotenlayouts

Das Erscheinugsbild von Fußnoten kann durch folgende Parameter geändert werden:



Parameter	Bedeutung
`\footnotesize`	Schriftgröße der Fußnoten
`\footnotesep`	Abstand zwischen zwei Fußnoten
`\skip\footins`	Ein TeX-Kommando, das den Abstand der Fußnote zum Hauptext festlegt. Veränderung mit `\addtolength{\skip\footins}{`*dim*`}`.
`\footnoterule`	Erzeugt den Trennungstrich zwischen Haupttext und der Fußnote. Kann mittels `\renewcommand` geändert werden: `\renewcommand{\footnoterule}{\vspace*{-3pt}%` `\rule{7em}{.4pt}\vspace*{2pt}}`

Das Fußnotensymbol kann durch Umdefinition des `\@makefnmark`-Befehls verändert werden.

Beispiel:

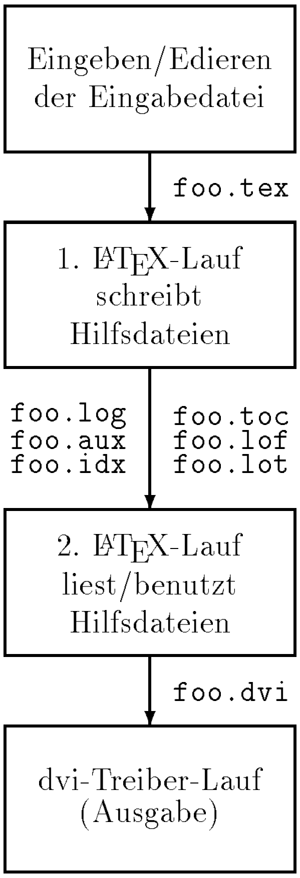
```
\renewcommand{\@makefnmark}{\mbox{$^{(\@thefnmark)}$}}
```

erzeugt ein Fußnotensymbol der Form $^{(3)}$.

Die Erzeugung des Fußnotentextes lautet in der Voreinstellung:

```
\newcommand{\@makefntext}[1]%
 {\noindent\makebox[1.8em][r]{\@makefnmark}#1}
```

Inhaltsverzeichnis



Die *Gliederungsbefehle* (`\section`, `\subsection`, usw.; siehe Folie 2.3.2) prägen das Layout eines Dokuments entscheidend mit. Darüberhinaus können die Gliederungsbefehle gleich die notwendigen Informationen zur automatischen Erzeugung eines Inhaltsverzeichnisses mitliefern.

LaTeX schreibt bei der Bearbeitung der Datei `foo.tex` die Informationen über Gliederungsebenen, Überschriften und die betreffenden Seitenzahlen in eine Datei namens `foo.aux`.

Diese Datei könnte z.B. wie folgt aussehen:

```
\relax
\@writefile{toc}{\contentsline{section}
  {\numberline {0.1} Randnotizen,
  Fu\"snoten und Verzeichnisse}{2}}
\@writefile{toc}{\contentsline{subsection}
  {\numberline {0.1.1} Randnotizen}{2}}
```

- Mit dem Befehl `\tableofcontents` wird ein Inhaltsverzeichnis an der Stelle im Text eingefügt, wo der Befehl eingetragen ist.
- Entsprechend erzeugen die Befehle `\listoffigures` und `\listoftables` ein Abbildungs- bzw. Tabellenverzeichis für Objekte, die mit einer Abbildungs- oder Tabellenunterschrift (`\caption`) versehen sind.
- Die Verzeichniseinträge werden in Dateien mit Namen `foo.toc`, `foo.lof` und `foo.lot` eingetragen (für das Inhalts-, Abbildungs- und Tabellenverzeichnis respektive).
- Da die `.aux`-Datei erst nach dem ersten Durchlauf von LaTeX zur Verfügung steht, stimmt das Inhaltsverzeichnis erst nach dem zweiten LaTeX-Lauf mit der wahren Gliederung überein.
- Jede Verschiebung in der Gliederung macht zwei LaTeX-Läufe notwendig, um eine wahre Gliederung zu erhalten.

Ergänzungen der Verzeichnisse

Es können auch Informationen in ein Verzeichnis eingetragen werden, die nicht aus einem Gliederungsbefehl oder einer Bildunterschrift gewonnen werden.

- An das Verzeichnis `foo.`*file* kann mit dem Befehl

  `\addtocontentsline{`*file*`}{`*text*`}`

  der Text *text* angehängt werden.

  Beispiel:

  ```
  \addtocontentsline{toc}
  {\numberline{}Abschlie"sende
  Bemerkungen}
  ```

  Das Beispiel fügt an das Ende des Inhaltsverzeichnis (`foo.toc`) die entsprechende Zeile an.

- Mit der Setzung des Zähler `tocdepth` wird eingestellt, bis zu welcher Gliederungstiefe ein Verzeichnis aufgebaut werden soll. Mit dem Befehl

  `\setcounter{tocdepth}{2}`

  werden in der Dokumentklasse *book* z.B. nur noch die Einträge für Kapitel und Abschnitte in das Inhaltsverzeichnis aufgenommen. Normalerweise werden bei *book* drei Gliederungsebenen aufgenommen.

- Gliederungsbefehle mit angehängtem $*$ werden nicht numeriert und nicht in das Inhaltsverzeichnis aufgenommen (z.B. `\section*`). Möchte man eine mit einem solchen $*$-Befehl erzeugte Überschrift trotzdem ins Inhaltsverzeichnis eintragen lassen, kann man das z.B. folgendermaßen erreichen:

  ```
  \chapter*{Vorwort}
  \addtocontentsline{toc}
  {\numberline{}Vorwort}
  ```

Abschnitt 3

Absatzformatierung

Gibt das Layout der Seite den allgemeinen Rahmen für einen gedruckten Text vor, so „lebt“ dieser Text doch eigentlich in den Absätzen. Die Absätze grenzen Sinnzusammenhänge ein und können durch ihre Gestaltung das Wesen eines Abschnitts hervorheben: So werden Zitate anders gesetzt als normaler Fließtext, und Gedichte besitzen eine andere Form als die langen Tabellen einer Inventurliste. Im folgenden werden die technischen Hilfsmittel erläutert, die LaTeX bereitstellt, um solche Zusammenhänge angemessem durch die Art der Absatzformatierung zum Ausdruck zu bringen.

3.1 Grundlagen

3.1.1

Umgebungen

Umgebungen ermöglichen die Formatierung ganzer Textabschnitte. Sie stellen dazu spezielle Befehle bereit oder setzen bestimmte Parameter. Die Wirksamkeit aller Befehle in einer Umgebung ist auf den von ihr eingeschlossenen Textabschnitt eingeschränkt.

- Jede Umgebung besteht aus zwei Befehlen, einem Anfangsbefehl

  `\begin{`*name*`}`

  und einem Endbefehl

  `\end{`*name*`}`

  Die Anfangs- und Endbefehle arbeiten jeweils eine definierte Folge von LaTeX-Befehlen ab. Der Umgebungsname *name* muß in beiden Befehlen übereinstimmen.
- Der Anfangsbefehl kann weitere notwendige und ein optionales Argument besitzen.
- Zwischen Anfangs- und Endbefehl steht der durch die Umgebung zu formatierende Text.

Beispiel:

```
\begin{center}
Text
\end{center}
```

Text

- Umgebungen können (wie Klammern) geschachtelt werden. Dabei dürfen die Klammerungshierarchien nicht verzahnt werden.

Erlaubt ist also:

```
\begin{nameA}
   \begin{nameB}
      ...
   \end{nameB}
\end{nameA}
```

Verboten ist hingegen:

```
\begin{nameA}
   \begin{nameB}
      ...
   \end{nameA}
\end{nameB}
```

- Es können eigene Umgebungen definiert werden.

Direkte (unformatierte) Ausgabe

Manchmal soll der Quelltext genauso gesetzt werden, wie er eingegeben wurde, d.h. mit allen Leerzeichen und Zeilenwechseln und ohne Interpretation der LaTeX-Befehle.

Dies ist mit Hilfe der *verbatim*-Umgebung möglich.

Beispiel:

```
\begin{verbatim}
In der verbatim-Umgebung wird kein Zeilenumbruch
durchgef"uhrt: Die Zeilen enden dort, wo sie auch
im Editor enden. Es wird ein Schreibmaschinenfont
verwendet. \LaTeX-Befehle werden nicht expandiert.
\end{verbatim}
```

```
In der verbatim-Umgebung wird kein Zeilenumbruch
durchgef"uhrt: Die Zeilen enden dort, wo sie auch
im Editor enden. Es wird ein Schreibmaschinenfont
verwendet. \LaTeX-Befehle werden nicht expandiert.
```

Der Befehl `\verb` hat eine ähnliche Wirkung, erzeugt jedoch keinen neuen Absatz, und wird daher vor allem im laufenden Text verwendet:

`\verb:\input:`

ergibt `\input`

Besonderheit: Bei diesem Befehl ist die Argumentklammer frei wählbar und besteht aus dem Zeichen, das dem Befehlsnamen unmittelbar folgt.

Beispiel:

Erlaubt ist: `\verb@blau@` , `\verb:blau:` oder `\verb(blau(`

Nicht erlaubt ist: `\verb(blau)`

3.1.3

Block- und Flattersatz

LaTeX verwendet als Normalsatz den Blocksatz, bei dem der Wortzwischenraum so verändert wird, daß ein glatter linker und rechter Rand entsteht. Hierbei geht TeX *absatzharmonisch* vor, d.h. das Programm achtet auf möglichst gleichmäßige Wortabstände innerhalb eines Absatzes.

- Die Spaltenbreite für den normalen Fließtext ist bei einspaltigem Satz durch den `\textwidth` Parameter gegeben.

Es sind drei weitere Satzarten möglich: zentrierter, sowie links- oder rechtsbündiger Satz. Hierbei wird der natürliche Wortabstand verwendet. Bei Erreichen des Zeilenendes wird ohne Worttrennung eine neue Zeile begonnen.

- Die Umgebung *flushright* setzt den von ihr eingeschlossenen Text rechtsbündig.
- Die Umgebung *flushleft* setzt den Text linksbündig.
- Die Umgebung *center* setzt zentrierten Text.

Beispiele:

```
\begin{flushleft}
Der Flattersatz\\
ist durch die mechanische
Schreibmaschine durchaus\\
bekannt,
\end{flushleft}

\begin{flushright}
rechtsb"undiger Flattersatz\\
dagegen nicht.
\end{flushright}

\begin{center}
F"ur den Titel eines\\
Dokuments wird\\
h"aufig zentrierter\\
Textsatz verwendet.
\end{center}
```

Der Flattersatz
ist durch die mechanische
Schreibmaschine
durchaus
bekannt,

rechtsbündiger
Flattersatz
dagegen nicht.

Für den Titel eines
Dokuments wird
häufig zentrierter
Textsatz verwendet.

- Mit dem Befehl \\ kann in allen Satzarten der Beginn einer neuen Zeile erzwungen werden.

Freie Satzarten

Satzartbefehle. Neben der *flushleft*-, *flushright*- und der *center*-Umgebung stehen die folgenden Satzartbefehle zur Verfügung:

Befehl	Bedeutung
`\raggedright`	ab hier linksbündig
`\raggedleft`	ab hier rechtsbündig
`\centering`	ab hier zentriert

Nach Absetzen eines dieser Befehle muß jede Zeile mit `\\` abgeschlossen werden. Im Unterschied zu den Umgebungen wirken die Satzartbefehle als Schalter. Die gewählte Satzart wird solange beibehalten, bis sie durch einen anderen Satzartbefehl beendet wird. Da kein solcher Befehl für den Blocksatz existiert, ist die Anwendung dieser Befehle nur innerhalb von Gruppen oder Umgebungen sinnvoll, da dort die Wirksamkeit jeden Befehls auf den Bereich der Gruppe oder Umgebung beschränkt ist. Einzelne Zeilen können mit dem Befehl `\leftline{`*Zeile*`}` linksbündig und mit `\rightline{`*Zeile*`}` rechtsbündig angeordnet werden.

Freie Satzformen. Neben den Standardsatzformen Block- und Flattersatz finden vor allem für bestimmte Literaturformen (z.B. Gedichte) und in der Werbung Satzarten eine Anwendung, die einen mehr oder weniger frei definierten linken und rechten Rand erfordern. Man sollte sich aber bewußt sein, daß diese Satzformen nicht für „normale" Texte geeignet sind, da sie die Lesbarkeit langer Dokumente erschweren. Auch hier sollte die Regel beherzigt werden, daß Form und Inhalt eines Dokuments möglichst übereinstimmen sollten. Im Zweifelsfall greift man besser zu einem einfachen und klaren Layout.

Der Befehl `\hangindent=`*maß* zieht den Text links um *maß* ein. Danach muß mit dem Befehl `\hangafter=`*num* erklärt werden, auf wieviele Zeilen sich dieser Einzug bezieht: Ist *num*< 0, so werden die ersten *num* Zeilen des nachfolgenden Absatzes eingerückt. Ist *num*> 0, so werden die ersten *num* Zeilen nicht eingerückt, sondern die danach (Hier wurde `\hangindent=2cm` und `\hangafter=-3` verwendet).

Dieser Absatz wurde mit dem *shapepar* Paket formatiert, das eine geometrische Satzgestaltung ermöglicht. Es gibt bereits vorgegebene Randformen (Raute, Herz, ...), frei definierte Ränder sind aber auch möglich. Ein solcher Absatz darf keine mathematischen Formeln, Fußnoten oder Befehle wie `\vspace` enthalten. Nähere Informationen enthält das Paket selbst.

Mit dem TeX-Befehl

`\parshape=`*num* $i_1\, l_1\, i_2\, l_2 \ldots i_n\, l_n$

lassen sich beliebige Absatzränder definieren. Dabei gibt *num* die Anzahl der zu formatierenden Zeilen des nächsten Absatzes an, die Paare $i_j l_j$ geben den Einzug bzw. die Zeilenlänge an. Zum Beispiel wurde mit

```
\parshape=8
    0.0\linewidth   1.0\linewidth
    0.07\linewidth  0.86\linewidth
    0.13\linewidth  0.73\linewidth
                 ⋮
    0.37\linewidth  0.21\linewidth
    0.41\linewidth  0.15\linewidth
```

die folgende altertümliche Art, eine Seite zu beenden, gesetzt. Diese Satzart wird *Spitzkolumne* genannt. Solche Abschlußkolumnen werden jedoch nicht mehr benutzt.

3.1.4

Wortzwischenräume und Durchschuß

Der Abstand zwischen Worten und Zeilen (Durchschuß) kann wie folgt geändert werden:

- `\hspace{`*dim*`}` läßt einen Abstand der Größe *dim* frei. Dieser Abstand kann auch negativ sein. Die Form `\hspace*{`*dim*`}` wirkt auch am Zeilenanfang oder Ende.

  Beispiel:

  ```
  Hier \hspace{2cm} ist 2 cm Abstand.
  ```

  Hier   ist 2 cm Abstand.

  0   2

- Durch `\vspace{`*dim*`}` wird der Durchschuß zwischen zwei Zeilen um die Größe *dim* vergrößert oder verkleinert.

  ```
  Hier

  \vspace{2cm}

  ist 2 cm Abstand.
  ```

  0

  2

  Hier

  ist 2 cm Abstand.

- Der Durchschuß für das gesamte Dokument wird mit dem Befehl

  `\renewcommand{\baselinestretch}{`*faktor*`}`

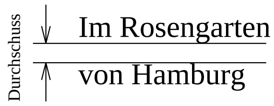
  geändert, wobei die Voreinstellung für *faktor* gleich `1.0` ist. Die Umschaltung wird erst nach dem nächsten Schriftgrößenbefehl wirksam.

- Wird der Zeilenumbruch durch `\\` erzwungen, kann zusätzlicher Durchschuß durch Anhängen von `[`*dim*`]` angegeben weden; z.B. `\\[.3ex]`.

- Für die Veränderung der Durchschußgröße werden elastische Maße, wie z.B. em oder ex empfohlen, da sie sich der jeweiligen Schriftgröße automatisch anpassen. Mit dem Abstand `\fill` wirken `\hspace` und `\vspace` als „Spreizen“: `A \hspace{\fill} B` positioniert z.B. die Buchstaben `A` und `B` ganz links und ganz rechts am Rand der Seite. Diese Befehle können mit `\hfill` und `\vfill` abgekürzt werden.

Anmerkungen zum Durchschuß

Werden beim Bleisatz die Zeilen direkt übereinander gesetzt, so entsteht ein sog. *kompresser* Satz. Ein solches Satzbild wirkt sehr kompakt und geschlossen, ist aber nur schwer lesbar. Deswegen wird im allgemeinen zusätzliches vertikales Material zwischen die Zeilen eingebracht, der sog. *Durchschuß*. Ein typographischer Richtwert [3] besagt, daß eine optimale Lesbarkeit bei etwa einem Durchschuß von einem Drittel der Schrifthöhe erreicht wird. Da im elektronischen Textsatz mit LaTeX keine soliden Lettern aus Blei existieren, wird der Durchschuß als der vertikale Raum zwischen der Schriftunterkante und der Schriftoberkante definiert:



Ein `\baselinestretch`-Faktor (bls-Faktor) von 1.0 entspricht in etwa einem Drittel Durchschuß. Eine Verringerung des bls-Faktors auf 0.9 erzielt die Wirkung eines kompressen Satzes. Hierbei können jedoch Probleme mit den diakritischen Zeichen der Großbuchstaben (z.B. Ü) auftreten, die zu optischen Unregelmäßigkeiten im Satzbild führen. Ein bls-Faktor von 1.1 kann zur Verbesserung der Lesbarkeit sinnvoll sein:

LaTeX verwendet als Normalsatz den Blocksatz, bei dem der Wortzwischenraum so verändert wird, daß ein glatter linker und rechter Rand entsteht. Hierbei geht TeX *absatzharmonisch* vor, d.h. das Programm

bls-Faktor = 0.9

LaTeX verwendet als Normalsatz den Blocksatz, bei dem der Wortzwischenraum so verändert wird, daß ein glatter linker und rechter Rand entsteht. Hierbei geht TeX *absatzharmonisch* vor, d.h. das Programm

bls-Faktor = 1.0

LaTeX verwendet als Normalsatz den Blocksatz, bei dem der Wortzwischenraum so verändert wird, daß ein glatter linker und rechter Rand entsteht. Hierbei geht TeX *absatzharmonisch* vor, d.h. das Programm

bls-Faktor = 1.1

Zuweilen wird für Publikationen und Abschlußarbeiten ein „$1\frac{1}{2}$ zeiliger“ Abstand verlangt. Dies darf nicht mit dem Durchschuß verwechselt werden. Vielmehr meint diese Forderung, daß der Abstand zwischen den Grundlinien dem $1\frac{1}{2}$-fachen der Entwurfsgröße der Schrift entspricht.

Daraus ergibt sich folgende Umrechnungstabelle (nach [4]) für den bls-Faktor:

Zeilen-Faktor	bls-Faktor bei Entwurfsgröße 10pt	11pt	12pt
$1\frac{1}{2}$	1.25	1.21	1.24
2	1.67	1.62	1.66

Zitieren längerer Abschnitte

In LaTeX wird die Textweite durch die Präambel für das gesamte Dokument festgelegt. Zitate werden dagegen oft über einen Einzug des linken und rechten Randes kenntlich gemacht. Dies ermöglichen die Umgebungen *quote* und *quotation*.

Beispiel:

```
Der Text in den Umgebungen \textsl{quote} und
\textsl{quotation} wird um den gleichen Betrag rechts
und links einger"uckt:

\begin{quote}
   Zwischen dem einger"uckten Text und dem Normaltext
   werden zus"atzliche Zwischenr"aume eingef"ugt.

   Abs"atze werden wie "ublich durch Leerzeilen
   getrennt.
\end{quote}

Mit \textit{quotation} wird die erste Zeile
eines Absatzes einger"uckt, mit \textit{quote} werden
die Abs"atze durch einen Zwischenraum getrennt.
```

Der Text in den Umgebungen *quote* und *quotation* wird um den gleichen Betrag rechts und links eingerückt:

> Zwischen dem eingerückten Text und dem Normaltext werden zusätzliche Zwischenräume eingefügt.
>
> Absätze werden wie üblich durch Leerzeilen getrennt.

Mit *quotation* wird die erste Zeile eines Absatzes eingerückt, mit *quote* werden die Absätze durch einen Zwischenraum getrennt.

3.1.6

Mehrspaltiger Satz

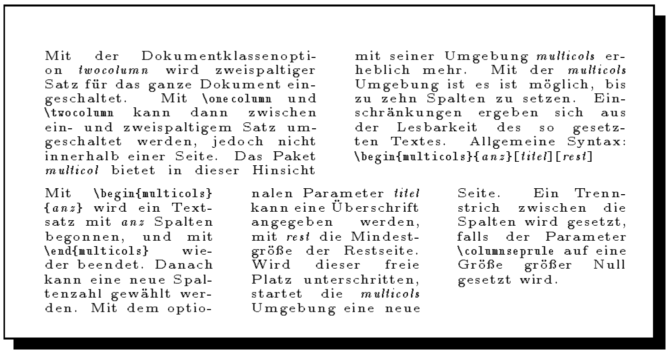
Das Paket *multicol* ermöglicht den Satz mehrspaltiger Texte:[1]

- Mit der *multicols*-Umgebung des Pakets ist es ist möglich, bis zu zehn Spalten zu setzen. Allgemeine Syntax:

  `\begin{multicols}{`*anz*`}[`*titel*`][`*rest*`]`

- Mit `\begin{multicols}{`*anz*`}` wird ein Textsatz mit *anz* Spalten begonnen, und mit `\end{multicols}` wieder beendet. Danach kann eine neue Spaltenzahl gewählt werden.

  Beispiel:



Mit der Dokumentklassenoption *twocolumn* wird zweispaltiger Satz für das ganze Dokument eingeschaltet. Mit \onecolumn und \twocolumn kann dann zwischen ein- und zweispaltigem Satz umgeschaltet werden, jedoch nicht innerhalb einer Seite. Das Paket *multicol* bietet in dieser Hinsicht mit seiner Umgebung *multicols* erheblich mehr. Mit der *multicols* Umgebung ist es ist möglich, bis zu zehn Spalten zu setzen. Einschränkungen ergeben sich aus der Lesbarkeit des so gesetzten Textes. Allgemeine Syntax: \begin{multicols}{*anz*}[*titel*][*rest*]

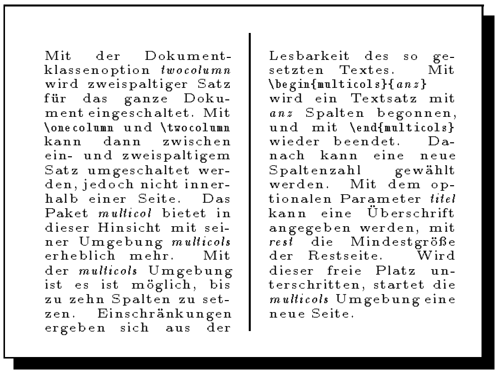
Mit \begin{multicols}{*anz*} wird ein Textsatz mit *anz* Spalten begonnen, und mit \end{multicols} wieder beendet. Danach kann eine neue Spaltenzahl gewählt werden. Mit dem optionalen Parameter *titel* kann eine Überschrift angegeben werden, mit *rest* die Mindestgröße der Restseite. Wird dieser freie Platz unterschritten, startet die *multicols* Umgebung eine neue Seite. Ein Trennstrich zwischen die Spalten wird gesetzt, falls der Parameter \columnseprule auf eine Größe größer Null gesetzt wird.

```
\begin{multicols}{2}
    :
\end{multicols}
\begin{multicols{3}
    :
\end{multicols}
```

- Mit dem optionalen Parameter *titel* kann eine Überschrift angegeben werden, mit *rest* die Mindestgröße der Restseite. Wird dieser freie Platz unterschritten, startet die *multicols* Umgebung eine neue Seite.

- Ein Trennstrich zwischen die Spalten wird gesetzt, falls der Parameter `\columnseprule` auf eine Größe größer Null gesetzt wird.

  Beispiel:



Mit der Dokumentklassenoption *twocolumn* wird zweispaltiger Satz für das ganze Dokument eingeschaltet. Mit \onecolumn und \twocolumn kann dann zwischen ein- und zweispaltigem Satz umgeschaltet werden, jedoch nicht innerhalb einer Seite. Das Paket *multicol* bietet in dieser Hinsicht mit seiner Umgebung *multicols* erheblich mehr. Mit der *multicols* Umgebung ist es ist möglich, bis zu zehn Spalten zu setzen. Einschränkungen ergeben sich aus der Lesbarkeit des so gesetzten Textes. Mit \begin{multicols}{*anz*} wird ein Textsatz mit *anz* Spalten begonnen, und mit \end{multicols} wieder beendet. Danach kann eine neue Spaltenzahl gewählt werden. Mit dem optionalen Parameter *titel* kann eine Überschrift angegeben werden, mit *rest* die Mindestgröße der Restseite. Wird dieser freie Platz unterschritten, startet die *multicols* Umgebung eine neue Seite.

```
\setlength{\columnseprule}%
             {.4pt}
\begin{multicols}{3}
 Mit der Dokument ...
 ... gesetzt wird.
\end{multicols}
```

[1] Die Dokumentklassenoption *twocolumn* ist nur eingeschränkt zu empfehlen, da sie lediglich zweispaltigen Satz ermöglicht und innerhalb einer Seite keinen Wechsel zwischen ein- und mehrspaltigem Satz zuläßt.

3.1.7

Listen

Listen bestehen aus Textabschnitten (*Items*), die durch Symbole, Nummern oder Begriffe gegliedert sind. Es stehen drei Standardlisten zur Auswahl: die Umgebungen *itemize*, *enumerate* und *description*.

- Innerhalb dieser Umgebungen wird ein Item mit dem Befehl `\item` erzeugt.

- Die Listenumgebungen unterscheiden sich durch die verwendeten Items:
  - Die Umgebung *enumerate* numeriert die Textabschnitte. Die Numerierung erfolgt automatisch.
  - *itemize* kennzeichnet die Textabschnitte mit Symbolen wie • oder $*$.
  - *description* gliedert die Textabschnitte mit frei wählbaren Begriffen oder Symbolen.

- Listen können in vier Ebenen (1 bis 4) geschachtelt werden.

  Beispiel:

```
\begin{enumerate}
\item Eine Aufz"ahlung.
   \begin{enumerate}
   \item Ein Unterpunkt.
   \item Noch ein einer.
   \end{enumerate}
\item Die Numerierung
erfolgt automatisch.
\end{enumerate}
```

1. Eine Aufzählung.
   (a) Ein Unterpunkt.
   (b) Noch ein einer.
2. Die Numerierung erfolgt automatisch.

- Einrückung und Bezeichnung der Items wechseln mit jeder Ebene automatisch.
- Bei der *description* Umgebung muß die gewünschte Marke dem `\item`-Befehl als optionales Argument übergeben werden.

  Beispiel:

```
\item[Dies ist mein Item]
```

Standardlisten im Detail

Für die Standardlisten *enumerate* und *itemize* erfolgt die Wahl und Fortzählung der Marken der Items automatisch. Dies wird erreicht, indem LaTeX spezielle Zähler einrichtet. Für die *itemize*-Umgebung sind dies die Zähler `itemi`, `itemii`, `itemiii` und `itemiv`, für die *enumerate*-Umgebung die Zähler `enumi`, `enumii`, `enumiii` und `enumiv`. Jeder dieser Zähler hat eine bestimmte Voreinstellung, die der folgenden Tabelle entnommen werden kann.

Umgebung	Item autom.	Ebene	1	2	3	4
itemize	ja	Zähler:	`itemi`	`itemii`	`itemiii`	`itemiv`
		Beispiel:	•	–	∗	·
enumerate	ja	Zähler:	`enumi`	`enumii`	`enumiii`	`enumiv`
		Beispiel:	1., 2., ...	(a), (b), ...	i., ii., ...	A., B., ...
description	nein		frei wählbares Item auf allen Ebenen			

Soll ein einzelnes Item abweichend von der Voreinstellung gesetzt werden, so kann dies im optionalen Argument des `\item` Befehls angegeben werden. Zum Beispiel erzeugt man in der *itemize*-Umgebung mit dem Befehl

```
\item[+]
```

die Marke + anstelle der Marke • in der ersten Ebene. In der *description*-Umgebung *müssen* alle Marken auf diese Weise angegeben werden.

Soll das Erscheinungsbild der Marken einer Standardliste für das gesamte Dokument geändert werden, so ist dies durch Redefinition des `\label`*cnt*-Befehls möglich. Mit diesem Befehl erzeugt LaTeX intern die Marken der Standardlisten. Für *cnt* ist dabei einer der oben beschriebenen Zähler einzusetzen. Eine Änderung der Marken kann dann mit dem Befehl

`\renewcommand{\label`*cnt*`}{`*def*`}`

vorgenommen werden, wobei *def* die Definition der Marke ist, die anstelle der Standardmarke verwendet werden soll. Zum Beispiel wird mit der folgenden Definition der Stil der Zähler der *enumerate*-Umgebung geändert:

```
\renewcommand{\labelenumi}{\Alph{enumi})}
\renewcommand{\labelenumii}{\alph{enumii}.)}
\renewcommand{\labelenumiii}{\roman{enumiii}.)}
\renewcommand{\labelenumiv}{\labelitemi}
```

Die Marken erscheinen dann so:

A) Ebene eins.
  a.) Ebene zwei.
    i.) Ebene drei.
      • Ebene vier.

3.2 Listen und Boxen

3.2.1

Selbstdefinierte Listen

Listen vom Typ *itemize*, *enumerate* und *description* lassen sich in einer allgemeinen Form vom Benutzer gestalten. Die Syntax lautet:

\begin{list}{*Standardmarke*}{*Listenerklärung*}
  Itemliste
\end{list}

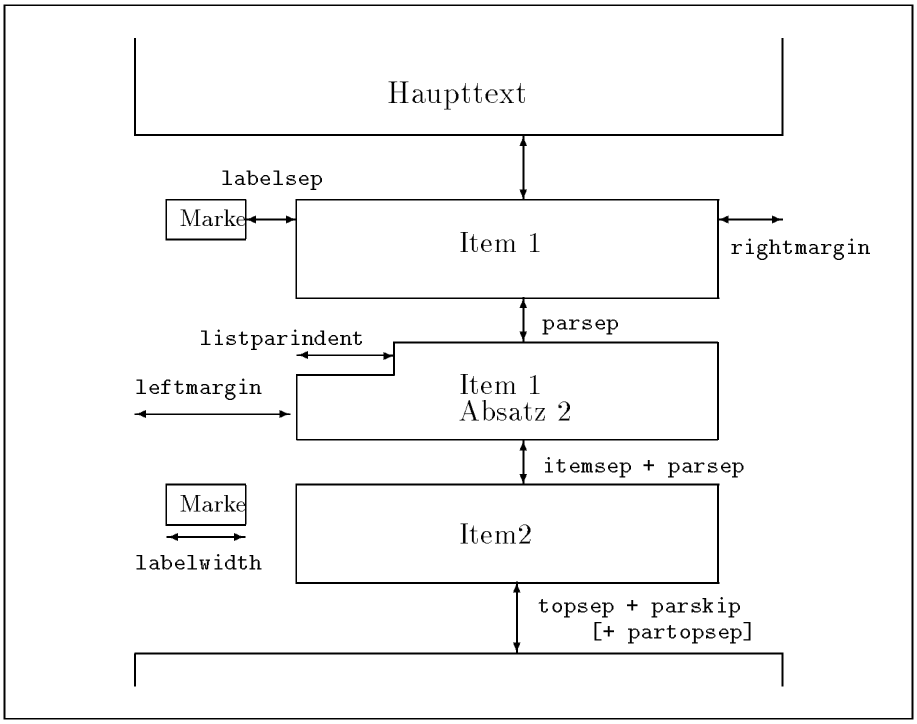
Argument I:	*Standardmarke*. Diese wird beim `\item` - Befehl ausgedruckt. Für eine fortlaufende Numerierung muß ein Zähler eingerichtet werden.
Argument II:	*Listenerklärung*. Setzung einiger oder aller verfügbaren Listenparameter auf vom Benutzer gewünschte Werte, welche die Anordnung der Liste beeinflussen. Bei Nichtangabe werden die Voreinstellungen von LaTeX übernommen.
Body:	*Itemliste*. Hier folgt die Liste der Items, beginnend mit dem `\item` - Befehl.

Diese Liste wurde so erzeugt:

```
\newcounter{fig}
\begin{list}{ Argument \Roman{fig}:}{ \usecounter{fig}%
  \labelwidth7em \labelsep1.5em \leftmargin7em}
\item \textit{Standardmarke}. Diese wird beim
   \verb|\item| - Befehl ausgedruckt. F"ur eine
   fortlaufende Numerierung ...
\item \textit{Listenerkl"arung}. Setzung einiger oder
   aller verf"ugbaren Listenparameter auf vom Benutzer
   gew"unschte Werte, welche  ...
\item[Body:] \textit{Itemliste}. Hier folgt die Liste der
   Items, beginnend mit dem \verb|\item| - Befehl.
\end{list}
```

Ändern des Listen-Layouts

Das Listen-Layout kann mit folgenden Parametern beeinflußt werden:



Listenparameter	Bedeutung
`\topsep`	Vertikaler Zwischenraum zwischen Text und Liste zusätzlich zu `\parskip`.
`\partopsep`	Zusätzlicher vertikaler Zwischenraum zwischen erstem oder letztem Item und angrenzendem Text (wenn durch Leerzeile getrennt).
`\parsep`	Vertikaler Abstand zwischen Absätzen *eines* Items.
`\itemsep`	Abstand zwischen Items zusätzlich zu `\parsep`.
`\leftmargin`	Einrücktiefe links relativ zum Rand der aktuellen Umgebung.
`\rightmargin`	Einrücktiefe rechts relativ zum Rand der aktuellen Umgebung.
`\listparindent`	Zusätzliche Einrücktiefe der ersten Zeile eines neuen Absatzes im Item (Standard: 0cm).
`\labelwidth`	Breite des Markierungsfeldes.
`\labelsep`	Abstand zwischen Markierung und Listentext.

Definition eigener Umgebungen

Neue Umgebungen können in LaTeX folgendermaßen selbst definiert werden:

`\newenvironment{`*cmd*`}[`#`][`*default*`]{`*b-def*`}{`*e-def*`}`

- *cmd* ist der neue Umgebungsname, den der Benutzer frei wählen kann, soweit dieser nicht bereits definiert ist.
- *b-def* und *e-def* sind die Befehlsfolgen, die durch den Anfangs- bzw. Endbefehl der Umgebung abgearbeitet werden.

  Beispiel:

  ```
  \newenvironment{mitte}{\begin{center}}{\end{center}}
  ```

- Soll eine selbstdefinierte Umgebung Argumente besitzen, so wird deren Anzahl mit der Zahl # festgelegt. Auf diese Argumente kann in *b-def* und *e-def* mit den Symbolen #1, #2, ... (für das erste, zweite, usw. Argument) zugegriffen werden.
- Wird in der Definition das optionale Argument *default* angegeben, kann ein optionales Argument beim Aufruf der neuen Umgebung verwendet werden. Dieses erhält das Symbol #1. Bei Aufruf der neuen Umgebung ohne optionales Argument, liefert dann ein Zugriff auf Symbol #1 den Wert *default*.

  Beispiel:

  ```
  \newenvironment{vonbis}[2][1980]%
                 {\centerline{von #1 bis #2}}{\relax}
  ```

  ergibt bei Eingabe von

  ```
  \begin{vonbis}{1985}\end{vonbis}
  ```

  die Zeile

  von 1980 bis 1985

- Soll eine in LaTeX bereits definierte Umgebung verändert werden, so ist hierzu der Befehl `\renewenvironment` zu verwenden. Dieser Befehl hat die gleiche Syntax wie `\newenvironment`.

Eigene Umgebungen mit Listen

Eine Listendefinition kann in einem selbstdefinierten Environment festgehalten werden. Nach der Definition

```
\newenvironment{quademize}{%
      \begin{list}{\Large$\square$}{%
      \labelwidth2em%
      \leftmargin4.5em \labelsep1.5em \bfseries%
      }%
  }{\end{list}}
```

kann die Umgebung *quademize* wie folgt benutzt werden:

```
\begin{quademize}
\item Diese Liste besitzt eine sehr auff"allige
  Markierung.
\item Es wird ein fetter Font gew"ahlt.
\item Diese Liste kann nicht (vern"unftig)
  geschachtelt werden.
\end{quademize}
```

- ☐ **Diese Liste besitzt eine sehr auffällige Markierung.**
- ☐ **Es wird ein fetter Font gewählt.**
- ☐ **Diese Liste kann nicht (vernünftig) geschachtelt werden.**

3.2.5

Boxen und Stützelemente

LaTeX behandelt alle Textelemente unterschiedslos als „Boxen“, d.h. als Objekte mit bestimmter Breite, Höhe über der Grundlinie und Tiefe unter der Grundlinie. Es stehen drei verschiedene Boxentypen für eigene Konstruktionen zur Verfügung:

- Rule-Boxen erzeugen Linien und ausgefüllte Rechtecksflächen.
- LR-Boxen erlauben nur die horizontale Anordnung von Textelementen. Es findet kein Zeilenumbruch statt.
- Paragraphenboxen ermöglichen einen Zeilenumbruch.

Mit Boxen lassen sich auch komplizierte Layoutvorstellungen verwirklichen:

- Zeilenfreies Positionieren von Text, auch über den Satzspiegel hinaus.
- Nebeneinanderstellen von Textabschnitten und Übereinanderdrucken von Graphik und Text.
- Erzeugen unsichtbarer Stützelemente. Diese sind ein wichtiges Hilfsmittel, um „Platz zu schaffen“. Sie können mit Hilfe der Rule-Boxen verwirklicht werden:
  - Syntax:
    `\rule[`*lift*`]{`*breite*`}{`*höhe*`}`
  - Mit dem optionalen Argument *lift* kann die Rule-Box über (oder bei negativem Argument unter) die Grundlinie gesetzt werden, Breite und Höhe der Rule-Box müssen immer angegeben werden.
  - Wird entweder Breite oder Höhe einer Rule-Box auf 0cm gesetzt, so entsteht eine unsichtbare vertikale oder horizontale Linie (eine sog. Stütze).

Beispiel:

```
\rule{5.5cm}{.4pt}\\
\rule[-4mm]{0cm}{1.2cm}
\textsc{Johannes Gutenberg}\\
\rule{5.5cm}{.4pt}
```

JOHANNES GUTENBERG

3.2.6

LR-Boxen

LR-Boxen werden mit folgenden Befehlen erzeugt:

\mbox{*Text*}
\fbox{*Text*}

\makebox[*breite*][*pos*]{*Text*}
\framebox[*breite*][*pos*]{*Text*}

In der zweiten Form können die Boxbreite *breite* und ein Positionierungsparameter *pos* = `l` oder `r` angegeben werden. Mit `l` wird der Text in der Box links angeordnet, mit `r` rechts und ohne Parameter wird er zentriert.

Beispiele:

```
\fbox{Umrahmter Text}
```

Umrahmter Text

```
\framebox[8cm][r]{rechtsb"undiger Text}
```

rechtsbündiger Text

```
\fbox{\rule{0cm}{3cm}Text}
\fbox{\rule[-5mm]{0cm}{3cm}Text}
\fbox{\rule[-10mm]{0cm}{3cm}Text}
\fbox{\rule[-25mm]{0cm}{3cm}Text}
```

Text Text Text Text

3.2.7

Paragraphen-Boxen

In den Parboxen findet automatischer Zeilenumbruch statt. Paragraphen-Boxen können erzeugt werden mit

`\parbox[`*pos*`]{`*breite*`}{`*Text*`}`

oder

```
\begin{minipage}[pos]{breite}
            :
\end{minipage}
```

- Es kann ein Positionsparameter *pos* = `b` oder `t` angegeben werden: mit `t` wird die oberste Zeile der Box zum laufenden Text ausgerichtet, mit `b` die unterste. Ohne Positionierungsparameter wird die Box vertikal zur laufenden Zeile zentriert.
- Die Breite der Box muß mit *breite* angegeben werden.

In einer *minipage* Umgebung dürfen alle Befehle verwendet werden; sie kann also auch zentrierten Text, Listen, Tabulatoren ... enthalten. In einer `\parbox` sind diese Befehle nicht erlaubt.

Beispiel:

```
\parbox{5cm}{Hier steht ein einfaches ...}
%
\hfill Zeile \hfill
%
\fbox{\begin{minipage}[t]{6cm}
Boxen k"onnen auch geschachtelt ...
\end{minipage}}
```

Hier steht ein einfaches Beispiel einer 5 cm breiten Parbox	Zeile	Boxen können auch geschachtelt werden, um z.B. zusätzlich einen Rahmen zu erzeugen – hier z.B. um eine „minipage“.

3.3 Tabellen

3.3.1

Tabulatoren

Tabulatoren teilen eine Druckseite (unsichtbar) in Spalten auf, an denen der Text angeordnet werden kann. LaTeX bietet mit der *tabbing*-Umgebung die Möglichkeit, solche Tabulatoren zu setzen.

- In der *tabbing*-Umgebung kann ein Tabulator mit dem Befehl \= gesetzt werden. Jede Zeile ist explizit mit dem Befehl \\ abzuschließen.

  Beispiel:

  ```
  Der erste Tabulator steht \= hier.\\
  ```

- Mit dem Befehl \> kann auf die Tabulatorposition vorgerückt werden.

  Beispiel:

  ```
  \begin{tabbing}
  Der erste Tabulator steht \= hier.\\
  Dort \> springen wir hin.
  \end{tabbing}
  ```

Der erste Tabulator steht	hier.
Dort	springen wir hin.

- Es können mehrere Tabulatoren gesetzt und mehrere in Folge übersprungen werden. Mit dem Befehl `\kill` ist es möglich, zunächst in einer Musterzeile Tabulatoren zu setzen und diese dann später entsprechend zu verwenden.

Beispiel:

```
\begin{tabbing}
\hspace*{1em}\=Geburtstag: \=\kill
Datensatz:\\
\> Name:       \> Gutenberg\\
\> Vorname:    \> Johannes\\
\> Geburtstag: \> um 1400
\end{tabbing}
```

Datensatz:		
	Name:	Gutenberg
	Vorname:	Johannes
	Geburtstag:	um 1400

Tabular: Spaltendeklaration und Zeilenaufbau

In einer Tabelle wird Textmaterial in Zeilen und Spaltenform angeordnet. Oft werden auch horizontale und vertikale Linien zur optischen Trennung als Zeilen- bzw. Spaltentrenner eingesetzt.

- Diese Textsatzaufgaben lassen sich mit der Umgebung *tabular* durchführen.

  Beispiel:

a	b	c
AA	BB	CC
123	456	789

```
\begin{tabular}{ccc}
a    & b   & c   \\
AA   & BB  & CC  \\
123  & 456 & 789 \\
\end{tabular}
```

- Eine Tabelle wird durch Spaltendeklaration, Spaltenbegrenzer und Zeilenend-Befehle gegliedert:

```
\begin{tabular}{ccc}     Spaltendeklaration
a    & b   & c   \\
AA   & BB  & CC  \\        Zeilenende
123  & 456 & 789 \\
\end{tabular}
```

Spaltenbegrenzer

- Jeder Buchstabe in der Spaltendeklaration legt das Layout einer Spalte fest.
- Das Tabellenmaterial wird mit dem Zeichen & in Spalten eingeteilt. Die Anzahl der so eingeteilten Spalten muß mit der Anzahl der deklarierten Spalten übereinstimmen.
- Durch den Befehl \\ wird eine neue Zeile der Tabelle begonnen.

Vor der Spaltendeklaration kann mit einem optionalen Argument die vertikale Positionierung analog zu den Paragraphenboxen beeinflußt werden (vergleiche Folie 3.2.7).

3.3.3

Tabular: Spaltendeklaration im Detail

Die Spaltendeklaration der *tabular*-Umgebung legt die Formatierung jeder einzelnen Spalte einer Tabelle fest. Es gibt folgende Möglichkeiten:

<table>
<tr><th>decl</th><th>Bedeutung</th></tr>
<tr><td><code>l</code></td><td>Inhalt der Spalte erscheint linksbündig.</td></tr>
<tr><td><code>c</code></td><td>Inhalt der Spalte erscheint zentriert.</td></tr>
<tr><td><code>r</code></td><td>Inhalt der Spalte erscheint rechtsbündig.</td></tr>
<tr><td><code>p{</code>dim<code>}</code></td><td>Der Text wird in einer Parbox der Breite dim gesetzt (z.B. <code>p{5cm}</code>). Die erste Zeile der Parbox wird auf die laufende Tabellenzeile ausgerichtet.</td></tr>
<tr><td>|</td><td>Ein vertikaler Strich wird zwischen die vorhergehende und die folgende Spalte gesetzt.</td></tr>
<tr><td>||</td><td>Wie oben, jedoch wird ein Doppelstrich verwendet.</td></tr>
<tr><td>∗{num}{decl}</td><td>Die Spaltendeklaration decl wird num-mal wiederholt. Zum Beispiel ist <code>∗{3}{|r}</code> gleichbedeutend mit <code>|r|r|r</code>.</td></tr>
<tr><td><code>@{</code>text<code>}</code></td><td>Das Textelement wird wie bei den Trennstrichen zwischen die vorhergehende und folgende Spalte (in jeder Zeile!) gesetzt. Dabei wird der übliche Spaltenzwischenraum entfernt.</td></tr>
</table>

Beispiel:

```
\begin{tabular}{l|c|r@{,}l%
        @{\hspace{2mm}$\cal DM$\hspace{2mm}}||p{6cm}}
Hemd   & $1/2$ Arm &  60 & -- & Baumwolle; wei"s \\
Hose   &           & 159 & -- & Jeans; blau\\
Socken & Paar      &   7 & 98 & Wolle; wei"s, schwarz,
                                braun und blau
\end{tabular}
```

Hemd	1/2 Arm	60,–	$\mathcal{DM}$	Baumwolle; weiß
Hose		159,–	$\mathcal{DM}$	Jeans; blau
Socken	Paar	7,98	$\mathcal{DM}$	Wolle; weiß, schwarz, braun und blau

3.3.4

Tabular: Komplexe Tabellen

Oft besteht die Notwendigkeit, mehrere Spalten innerhalb einer Tabelle zusammenzufassen.

- Der Befehl

  `\multicolumn{`*anz*`}{`*decl* `}{`*inh*`}`

  - faßt *anz* Spalten zu einer Spalte zusammen,
  - formatiert diese Spalte gemäß der Deklaration *decl* und
  - setzt in dieser Spalte den Text *inh*.
- Da eine mit `\multicolumn` erzeugte Spalte *anz* Spalten belegt, ist die Anzahl der Spaltentrenner & entsprechend zu verringern.
- Der Befehl `\multicolumn` kann auch dazu dienen, eine Spalte in einer einzelnen Zeile abweichend von der Hauptdeklaration zu formatieren.
- Eine horizontale Linie wird mit dem Befehl `\hline` erzeugt. Soll die Linie nur von der *n*-ten bis zur *m*-ten Spalte reichen, ist der Befehl `\cline{`*n*-*m*`}` zu verwenden.

Beispiel:

```
\begin{tabular}{|l|r|r|r|}
\hline
        & \multicolumn{3}{c|}{Durchschn.~Umsatz in DM} \\
Filiale & \multicolumn{1}{c}{1992}
        & \multicolumn{1}{c}{1993}
        & \multicolumn{1}{c|}{1994}     \\ \hline
T"ubingen & 450.000 & 510.600 & 509.000 \\ \cline{2-4}
Stuttgart & 631.000 & 680.000 & 723.000 \\ \hline
\end{tabular}
```

	Durchschn. Umsatz in DM		
Filiale	1992	1993	1994
Tübingen	50.000	510.600	509.000
Stuttgart	631.000	680.000	723.000

Ändern des Tabellen-Layouts

Die *tabular*∗-Umgebung. Mit dieser Standardumgebung können Tabellen mit einer vorgegeben Breite erzeugt werden. Die Umgebung wird wie folgt aufgerufen

```
\begin{tabular*}{breite}{decl}
      ⋮
\end{tabular*}
```

Dabei bezeichnet *breite* ein festes oder elastisches Maß und *decl* die übliche Spaltendeklaration. Eine typische Anwendung sind Tabellen, die über die ganze Seitenbreite reichen:

```
\begin{tabular*}%
      {\linewidth}{|l|c|c|}
      ⋮
\end{tabular*}
```

Das *tabularx*-Paket. Während die *tabular*∗-Umgebung den Spaltenzwischenraum verändert, damit die Tabelle eine vorgegebene Breite erreicht, verändert die *tabularx*-Umgebung aus dem gleichnamigen Paket die Spaltenbreite. Dazu wird ein neuer Spaltendeklarator `X` eingeführt. Bei Anwendung dieser Umgebung sind einige Besonderheiten zu beachten, für die auf [4, Abschnitt 5.3.5] verwiesen sei.

Neue Spaltendeklaratoren. Mit dem Befehl

```
\newcolumntype{dnam}[narg]{decl}
```

können eigene Spaltendeklaratoren definiert werden. Dabei bezeichnet *dnam* das Symbol für den gewünschten Deklarator, *narg* die Anzahl der möglichen Argumente und *decl* die Spaltendeklaration, die bei Anwendung von *dnam* ausgeführt wird. Die Syntax ist also zum `\newcommand` Befehl gleichwertig. Definiert man zum Beispiel `\newcolumntype{M}{>{$}c<{$}}`, so sind

```
\begin{tabular}{M}
```

und

```
\begin{tabular}{>{$}c<{$}}
```

gleichwertige Deklarationen (die `>{...}` Syntax ist auf Seite 86 erklärt).

Tabellenparameter. Mit den folgenden Parametern kann das Layout einer Tabelle geändert werden; diese Parameter können vor der betroffenen Tabelle mit dem `\setlength` Befehl modifziert werden.

Parameter	Bedeutung
`\tabcolsep`	Größe des halben Spaltenabstands.
`\arrayrulewidth`	Liniendicke für horizontale und vertikale Linien.
`\doublerulesep`	Der Abstand von Doppellinien:

Vertikale Linien variabler Breite. Wie oben zu sehen, kann mittels `\arrayrulewidth` die Linienbreite nur simultan für vertikale und horizontale Linien verändert werden. Mit einem „Trick“ lassen sich aber zumindest die vertikalen Linien einzeln definieren: Ersetzt man z.B. in der Deklaration `{|c|c|c|}` den zweiten Liniendeklarator auf folgende Weise (vgl. Seite 86):

```
\begin{tabular}%
      {|c!{\vrule width 2pt}|c|}
      ⋮
\end{tabular}
```

so wird die Tabelle wie folgt gesetzt:

eins	zwei	drei
vier	fünf	sechs

3.3.5

Fontauswahl in Tabellen

Sollen in einer Tabelle verschiedene Schriften verwendet werden, so gibt es dafür mehrere Möglichkeiten:

- Im Normalfall ist der Schriftwechsel für jeden Tabelleneintrag getrennt vorzunehmen, z.B.

```
\begin{tabular}{ccc}
\textit{Eins}  & {\small\texttt{zwei} & drei \\
  ⋮
```

- Das *array*-Paket bietet hier wesentlich bequemere Konstruktionen an. Das Paket wird mit

```
\usepackage{array}
```

  in der Präambel geladen. Tabellen können dann genauso gesetzt werden wie im Normalfall, jedoch werden folgende ergänzende Spaltendeklaratoren eingeführt:

Befehl	Bedeutung
`m`{*dim*}	Der Text wird vertikal (!) in einer Parbox der Breite *dim* zentriert (z.B. `m{5cm}`).
>{*text*}*decl*	Fügt den Text *text* in jeder Zeile an den Anfang der mit *decl* deklarierten Spalte ein (z.B: `>{\small}l`). Mögliche Deklaratoren sind `l,c,r,p,b,m`.
decl<{*text*}	Wie >, jedoch wird *text* an das Ende der durch *decl* deklarierten Spalte gesetzt.
`!`{*text*}	Wie `@{...}`, jedoch wird der Spaltenzwischenraum nicht unterdrückt.

- Mit der Deklaration `>{\itshape}l` wird die betreffende Spalte z.B. linksbündig in kursiver Schrift formatiert. So kann die Schriftart einer Tabelle spaltenweise festgelegt werden.

Gleitende Tabellen

Da die Standardtabellen in der *tabular*-Umgebung nicht über Seiten hinweg umgebrochen werden können, muß für jede Tabelle eine Stelle im Text gefunden werden, in die die Tabelle ganz hineinpaßt. Dies von Hand durchzuführen wäre nicht nur mühsam, sondern könnte auch leicht zu Verdruß führen, wenn sich eine einmal gefundene günstige Stelle durch kleine Änderungen im Text oder Layout verschiebt. Deshalb besteht die Möglichkeit, sich diese Arbeit von LaTeX abnehmen zu lassen; schließlich weiß das Programm am besten, wo Platz ist.

Die *table*-Umgebung. Verwirklicht wird die Idee mit der *table*-Umgebung. Hiermit wird ein sog. *floating body* (gleitendes Objekt) definiert:

```
\begin{table}[pos]
Hier kommt die Tabelle hin
\end{table}
```

Ohne optionales Argument sucht LaTeX die nächstgünstige Stelle, wo die Tabelle gesetzt werden kann. Mit dem Parameter *pos* kann die Wahl beeinflußt werden:

pos	Bedeutung
`h`	Die Tabelle erscheint nach Möglichkeit dort, wo sie im Quelltext steht.
`t`	Positioniert die Tabelle am Anfang der nächsten Seite (nach Möglichkeit).
`b`	Positioniert die Tabelle am Ende der nächsten Seite (nach Möglichkeit).
`p`	Sammelt die Tabellen nach Möglichkeit auf einer „page of float" am Ende des Kapitels.
`!`	Hebt alle Einschränkungen der Floatparameter (s.u.) auf.

Der nachfolgende Text wird entsprechend vorgezogen (siehe Abb. 3.1).

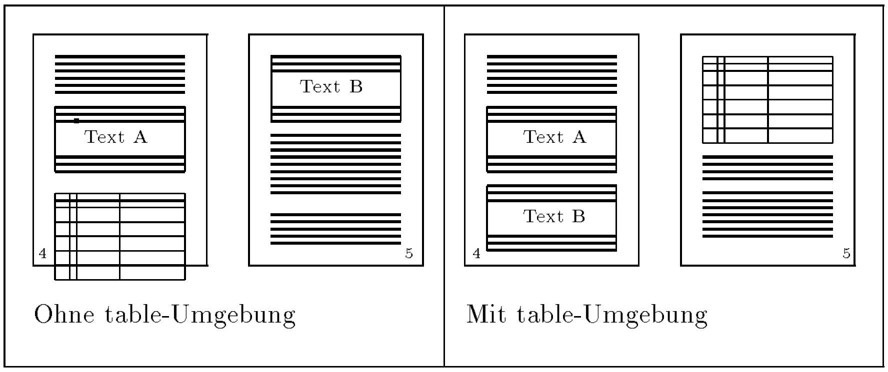


Abbildung 3.1: Eine gleitende Tabelle

Floatparameter. Die Anzahl der gleitenden Objekte pro Seite, sowie der Anteil von Text und gleitenden Objekten auf jeder Seite wird durch sog. *Floatparameter* bestimmt. So legt der Zähler `totalnumber` die maximale Anzahl der gleitenden Objekte pro Seite fest. Die Floatparameter können mit dem `\setcounter` bzw. dem `\setlength` Befehl gesetzt werden. Eine genaue Beschreibung findet sich auf Folie 5.2.2.

Überschriften. Jedes gleitende Objekt kann eine Überschrift oder Unterschrift mit dem Befehl `\caption[`*shortf*`]{`*longf*`}` erhalten. Dabei bezeichnet *longf* die Über- (Unter)schrift, *shortf* eine Kurzform, die in das Tabellenverzeichnis eingetragen wird. Der Befehl `\caption` muß in der *table*-Umgebung entsprechend vor oder hinter der Tabelle abgesetzt werden.

Tabular: Beispiel für das array-Paket

Diese Tabelle

Wichtig!	Sollen in einer Tabelle verschiedene Schriften verwendet werden, so gibt es dafür mehrere Möglichkeiten: Im Normalfall ist der Schriftwechsel für jeden Tabelleneintrag getrennt vorzunehmen.	hier endet *Zeile 1*
und	Das *array*-Paket bietet hier wesentlich bequemere Konstruktionen an.	hier endet *Zeile 2*

ist mit Hilfe des *array*-Pakets formatiert worden. Der Quelltext hat folgende Form:

```
\begin{tabular}{%
       >{\bfseries}l%
       >{\normalsize}m{4cm}!{hier endet}>{\itshape}l}
Wichtig! & Sollen in einer Tabelle verschiedene Schriften
           verwendet werden, so gibt es daf"ur mehrere
           M"oglichkeiten: Im Normalfall ist der
           Schriftwechsel f"ur jeden Tabelleneintrag
           getrennt vorzunehmen.
         & Zeile 1 \\
und      & Das \textit{array}-Paket bietet hier wesentlich
           bequemere Konstruktionen an.
         & Zeile 2
\end{tabular}
```

Das *supertab*-Paket

Bei der *tabular*-Umgebung muß der Umbruch einer langen Tabelle über Seiten hinweg von Hand vorgenommen werden. Diese Arbeit nimmt das *supertab* Paket dem Benutzer ab.

- Das Paket *supertab* stellt die *supertabular*-Umgebung bereit, mit der Tabellen gesetzt werden können, die sich über mehrere Seiten erstrecken. Das Paket wird mit `\usepackage{supertab}` in der Präambel geladen.
- Der Umbruch dieser Tabellen erfolgt automatisch. Es stehen die Spaltendeklaratoren der normalen *tabular*-Umgebung zur Verfügung.
- Es können Tabellenköpfe und Fortsetzungszeilen vereinbart werden, die automatisch beim Umbruch der Tabelle ausgewählt werden (siehe folgende Tabelle und das Beispiel auf Seite 90).

Befehl	Bedeutung
`\tablehead{...}`	Legt den Tabellenkopf fest, sofern `\tablefirsthead` nicht definiert ist. Der Tabellenkopf kann mehrere Zeilen enthalten (die mit `\\` getrennt werden).
`\tablefirsthead(...}`	Legt den Tabellenkopf der ersten Seite fest. Für die folgenden Seiten wird der in `\tablehead` definierte Kopf verwendet.
`\tabletail{...}`	Dieses Material wird an die letzte Tabellenzeile jeder Seite angehängt.
`\tablelasttail{...}`	Wie `\tabletail`, jedoch nur für die letzte Tabellenseite.
`\topcaption{...}`	Erzeugt eine Tabellenüberschrift, die in das Tabellenverzeichnis aufgenommen wird.
`\bottomcaption{...}`	Erzeugt eine Tabellenuntertitelung.

Die in der obigen Tabelle erklärten Setzungen müssen *vor* dem Umgebungsaufruf `\begin{supertabular}` abgesetzt werden (siehe Beispiel auf Seite 90). Die Setzungen für Kopf- und Fußzeilen müssen in der Spaltenanzahl mit der Tabellendeklaration übereinstimmen.

Obwohl die *supertabular*-Umgebung im allgemeinen ohne weitere Formatierungseingriffe gute Ergebnisse liefert, bestehen die folgenden Einschränkungen und Unverträglichkeiten:

- Eine Möglichkeit der vertikalen Ausrichtung bezüglich der laufenden Zeile (*tabular*: `b,t`) besteht nicht.
- Wenn „Float"-Objekte (wie die *figure*- oder *table*-Umgebung) auf der gleichen Seite definiert werden, kann es zu erheblichen Störungen in der Formatierung kommen („Zerreißen" des Layouts).

Beispiel für das *supertab*-Paket

Die folgende Tabelle zeigt, wie lange Tabellen über mehrere Seiten hinweg zu kontrollieren sind. Mit der Setzung von `\tabletail` wird bei Erreichen des Seitenendes die Tabelle unterbrochen und die Tabellenunterschrift „Fortsetzung auf der nächsten Seite“ gesetzt. Die Fortsetzung der Tabelle auf der folgenden Seite erhält die Überschrift „Fortsetzung der vorigen Seite“. Dies wird in `\tablehead` gesetzt.

Stil versus Klasse und Paket	
Dokumentstil	Dokumentklasse
`\documentstyle[`*Opt*`]{`*Stil*`}`	`\documentclass[`*Opt*`]{`*Klasse*`}`
Suffix: `.sty`	Suffix: `.cls`
article, report, book	article, report, book
letter	*nicht unterstützt*
externen Stylefiles	slides, ltxdoc, proc, ...
	Fortsetzung auf der nächsten Seite

—— Neue Seite ——

Fortsetzung der vorigen Seite	
Stylefile (Stiloption)	Paket
`\documentstyle[`*Style*`]{`*Stil*`}`	`\usepackage[`*Opt*`]{`*Paket*`}`

```
\tablefirsthead{%
    \hline \multicolumn{2}{|c|}{%
    Stil versus Klasse und Paket}\\
    \hline\hline}
\tablehead{\hline \multicolumn{2}{|l|}{%
    \small Fortsetzung der vorigen Seite}\\}
\tabletail{\hline \multicolumn{2}{r}{%
    \small Fortsetzung auf der n"achsten%
    Seite}\\}
\tablelasttail{}

\begin{supertabular}{p{6.8cm}|p{6.8cm}}
\multicolumn{1}{c|}{Dokumentstil}
    & \multicolumn{1}{c}{Dokumentklasse}\\ \hline
       ...
\end{supertabular}
```

Abschnitt 4

Mathematischer Formelsatz und Fonts

Ursprünglich insbesondere als System für anspruchsvollen mathematischen Formelsatz konzipiert, zeigt TEX gerade in diesem Bereich auch heute noch seine volle Stärke. Zwar verfügen inzwischen die meisten Textverarbeitungs- und Desktop-Publishing-Systeme auf WYSIWYG-Basis über Hilfsmittel zum Formelsatz (Formeleditoren, siehe z.B. [13]), doch ist in diesem Bereich ein vergleichbar flexibles System wie das Gespann TEX/LATEX — jedenfalls soweit den Autoren bekannt — nicht auf dem Markt zu finden.

Daß TEX im Formelsatz überragt, liegt nicht zuletzt an der Möglichkeit, TEX um eigene, qualitativ hochwertige Fonts zu erweitern, sollte tatsächlich einmal ein Sonderzeichen fehlen. Diese Möglichkeit wurde von Mathematikern auf aller Welt rege genutzt, und die zusammengetragenen Früchte dieser Arbeit lassen hinsichtlich der nunmehr verfügbaren Symbole kaum noch Wünsche offen.

4.1 Querverweise und Mathematischer Formelsatz

4.1.1

Querverweise

Um Bezüge auf andere Stellen im Text herzustellen, kann eine *unsichtbare* Marke gesetzt werden, auf die man später (oder auch *früher*) verweisen kann. Der Befehl

\label{*markierung*}

setzt solch eine Marke, wobei *markierung* aus beliebigen Zeichen bestehen darf[1]. Ein Verweis auf eine derartige Marke erfolgt mit einem der Befehle

\pageref{*markierung*}

oder

\ref{*markierung*}

Der \pageref-Befehl setzt die Seitennummer, auf der sich das entsprechende Label befindet, wogegen mit \ref{...} die „Bezugsnummer" des jeweils bezüglichen Textteils gesetzt wird. Textteile, die einen Bezug liefern können, sind in der folgenden Tabelle aufgeführt:

Objekt	Referenz
Gliederungsbefehl (\section)	Gliederungsnummer
equation, eqnarray	Gleichungsnummer
figure, table	Bild- oder Tabellennummer
enumerate	Wert des Items
theorem-Umgebung	entspr. Zählerstand

```
Diese Folie erh"alt ein Label\label{folie}.
Das Label \texttt{folie} steht auf Folie \ref{folie}.
```

Diese Folie erhält ein Label. Das Label `folie` steht auf Folie 4.1.1.

[1] Die Einzeichenbefehle \ # $ usw. sind nicht erlaubt.

4.1.2

Prinzipien beim Formelsatz

- Formelsatz erfordert einige Flexibilität, die bei „normalem“ Textsatz nicht notwendig ist. So können manche Elemente einer Formel erst bestimmt werden, wenn die Formel praktisch schon fertig gesetzt ist (z.B. Größe und Ausdehnung eines Wurzelzeichens über einem Formelausdruck).

  Beispiel:

  ```
  \sqrt{\frac{1+1}{(-2+1)\cdot(-1)}} = 1.414213\ldots
  ```

  $$\sqrt{\frac{1+1}{(-2+1)\cdot(-1)}} = 1.414213\ldots$$

- Somit ist es erforderlich, im Formelsatz graphische Elemente zu verwenden, deren Einsatz die Boxenlogik von TEX weit mehr strapaziert, als normaler Textsatz.

- LATEX verarbeitet Formelausdrücke unter anderem wegen solcher graphischer Aufgabenstellungen im sogenannten *mathematischen Modus* (im folgenden kurz MM) und stellt Umgebungen bereit, in denen man selbigen nutzen kann.

- Auch wenn der Formelgenerator von LATEX sehr leistungsstark ist, so kann er dennoch nicht alle Feinheiten, die beim Setzen einer Formel die Lesbarkeit fördern, automatisch berücksichtigen (zumal manche dieser Feinheiten stark vom Sinngehalt der Formel abhängen).

- Ein wichtiges Prinzip beim Formelsatz mit einem Textsystem lautet, die *Feinarbeit* an einer Formel erst unmittelbar vor dem endgültigen Layout vorzunehmen, da Formeln als graphische Elemente anzusehen sind und die Arbeit an graphischen Elementen bei z.B. einer Neustrukturierung des Textes erfahrungsgemäß die meiste Zeit in Anspruch nimmt.

Mathematische Umgebungen

Zum Setzen mathematischer Formeln dienen die folgenden drei Umgebungen (beziehungsweise fünf, je nach Zählweise):

- *math* plaziert eine Formel im Fließtext: ($i = \sqrt{-1}$) Die *math*-Umgebung ist auch abkürzend innerhalb von `\( ... \)` oder noch kürzer zwischen zwei Dollarzeichen `$ ... $` erreichbar.
- *equation* erzeugt eine abgesetzte und numerierte Formel:

$$e^{i\pi} + 1 = 0 \tag{4.1}$$

  Benutzt man *displaymath* (oder kürzer: `\[ ... \]`) an Stelle von *equation*, wird die Formel zwar abgesetzt, aber nicht numeriert.
- *eqnarray* erzeugt ein abgesetztes und numeriertes Gleichungsfeld:

$$\begin{aligned}
\pi &= a+b+c+d+e+f+g+h+i+ \\
&\quad j+k+l+m+n+o+p+q+r+ && (4.2) \\
&\quad s+t+u+v+w+x+y+z+\rho \\
a &= 3 \\
b &= 0.1 && (4.3) \\
c &= 0.04 \\
&\vdots \\
\rho &= Rest && (4.4)
\end{aligned}$$

  Ein nicht numeriertes Gleichungsfeld läßt sich mit Hilfe der Umgebung *eqnarray** anlegen.

  Die *eqnarray*-Umgebung ist wie eine dreispaltige Tabelle ohne Spaltendeklaration anzuwenden. Das obige Beispiel wurde so gesetzt:

```
\begin{eqnarray}
\pi & = & a + b + c + ... + z + \rho \\
a   & = & 3 \nonumber \\
b   & = & 0.1 \\
        :
        :
\rho& = & \mathit{Rest}
\end{eqnarray}
```

  Dabei dient der Befehl `\nonumber` zur Unterdrückung der Gleichungsnummerierung in dieser Zeile.

Mathematische Umgebungen: Besonderheiten

Die Umgebungen *math*, *equation*, *displaymath* und *eqnarray(*)* schalten LaTeX in den MM (mathematischen Modus). Im MM interpretiert LaTeX seine Eingaben auf eine für Formelsatz angepaßte Art und Weise:

- LaTeX schaltet auf einen mathematischen Italic-Zeichensatz um.
- LaTeX richtet das Spacing auf mathematische Symbole statt auf Wörter einer natürlichen Sprache aus (keine Ligaturen, kein Kerning).
- Daher dürfen im MM insbesondere keine Leerzeilen vorkommen!
- Leerzeichen im MM erzeugen später im Satz keinen Leerraum, sie sind hier nur als Kommandotrenner nötig (allerdings nach wie vor nützlich als optische Trenner für Übersichtlichkeit in der Eingabedatei).

Zur Erzeugung einer Tabelle mit beliebig vielen Spalten im MM[2] kann die *array*-Umgebung verwendet werden. Die *array*-Umgebung benötigt eine Spaltendeklaration, wie in der *tabular*-Umgebung und kann z.B. zur Darstellung von Matrizen benutzt werden.

Beispiel:

```
$
\left(
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
2 & 2 & 2 & 4 \\
3 & 2 & 1 & 4 \\
4 & 2 & 0 & 5
\end{array}
\right)
$
```

$$\left(\begin{array}{cccc}1 & 2 & 3 & 4 \\ 2 & 2 & 2 & 4 \\ 3 & 2 & 1 & 4 \\ 4 & 2 & 0 & 5\end{array}\right)$$

[2] Die *eqnarray*-Umgebung ist auf drei Spa;ten fixiert.

Grundlegende Befehle im MM

- Tiefstellen (Subscript): _{}: Durch Voranstellen des Unterstriches lassen sich Ausdrücke wie H_2O oder $G_{\mu\nu}$ setzen.

```
$H_2O$  $G_{\mu\nu}$
```

- Hochstellen (Superscript): ^{}: Ganz entsprechend wird durch Voranstellen eines Daches (Pfeil-nach-oben-Zeichen, ASCII 94) ein Ausdruck wie $r^2\pi$ oder $L^{\nu\mu}$ setzbar.

```
$r^2\pi$  $L^{\nu\mu}$
```

- Brüche (fractions): \frac{}{}: Ein Bruch wie $1/2$ ist einfach zu setzen, aber für $\frac{1}{2}$ braucht man einen Befehl.

```
$1/2$  $\frac{1}{2}$
```

- Wurzeln (roots): \sqrt[]{}: Ob nun eine Quadratwurzel wie $\sqrt{2}$ oder eine $n$te Wurzel wie $\sqrt[n]{x}$ gesetzt wird, man braucht in beiden Fällen den „squareroot“-Befehl.

```
$\sqrt{2}$  $n$te Wurzel  $\sqrt[n]{x}$
```

- Akzente (accents): Die folgenden Befehle erzeugen Akzente im MM:

$\hat{a}$	$\check{a}$	$\breve{a}$	$\acute{a}$	$\grave{a}$
\hat{a}	\check{a}	\breve{a}	\acute{a}	\grave{a}
$\tilde{a}$	$\bar{a}$	$\vec{a}$	$\dot{a}$	$\ddot{a}$
\tilde{a}	\bar{a}	\vec{a}	\dot{a}	\ddot{a}

Weitere Befehle im MM

- Zwei Extrabefehle stehen für „verlängerte Akzente“ zur Verfügung:

$\widehat{abc}$	$\widetilde{abc}$
`\widehat{abc}`	`\widetilde{abc}`

  Anmerkung: `\widevec` gibt es leider nicht.

- Horizontale Linien (over-/underlining): `\overline{}`: Einen `\widebar`-Befehl gibt es nicht, stattdessen nutzt man den „Überstreich“-Befehl:

```
Man kann $\overline{\mbox{"uberstreichen}}$
wie auch $\underline{\mbox{unterstreichen}}$.
```

  Man kann $\overline{\mbox{überstreichen}}$ wie auch $\underline{\mbox{unterstreichen}}$.

- Horizontale Klammern (over-/underbracing): `\overbrace{}`: Analog lassen sich horizontale geschweifte Klammern setzen (sogar mit Beschriftung, falls gewünscht):

```
\[
  \overbrace{h+h+\cdots+h}^{\mbox{i mal}} \qquad
  \underbrace{k+k+\cdots+k}_{\mbox{j mal}}
\]
```

$$\overbrace{h+h+\cdots+h}^{\mbox{i mal}} \qquad \underbrace{k+k+\cdots+k}_{\mbox{j mal}}$$

- Übereinanderstapeln (stacking): `\stackrel{}{}`: Dient dazu, zwei Ausdrücke im MM übereinander zu plazieren.

```
\pi \stackrel{?}{=} \frac{22}{7}
```

  $\pi \stackrel{?}{=} \frac{22}{7}$

4.1.7

Größenanpaßbare Operatoren und Begrenzer

- Größenanpaßbare Operatoren (sizable symbols): Es gibt Operatoren, deren Wirkung auf ganze Ausdrücke dazu führt, daß sich ihre Größe an diese Ausdrücke anpaßt (z.B. Integral- und Summen-Operator). Dies sind:

$\sum$	`\sum`	$\int$	`\int`	$\oint$	`\oint`	$\prod$	`\prod`
$\coprod$	`\coprod`	$\bigodot$	`\bigodot`	$\bigotimes$	`\bigotimes`	$\bigoplus$	`\bigoplus`
$\bigcap$	`\bigcap`	$\bigcup$	`\bigcup`	$\bigvee$	`\bigvee`	$\bigwedge$	`\bigwedge`
$\bigsqcup$	`\bigsqcup`	$\biguplus$	`\biguplus`				

Beispiel:

```
\[
  \int\limits_{-\infty}^\infty e^{-x^2}\,dx = \sqrt{\pi}
\]
```

$$\int\limits_{-\infty}^\infty e^{-x^2}\,dx = \sqrt{\pi}$$

- Begrenzer (delimiters): Begrenzer erfüllen die logische Funktion einer Klammer, weshalb sie paarweise auftreten müssen. Analog den obigen Operatoren passen die folgenden Begrenzersymbole ihre Größe einem eingeklammerten Ausdruck an, wenn man ihnen ein `\left` bzw. ein `\right` voranstellt:

$($	`(`	$\lfloor$	`\lfloor`	$\rfloor$	`\rfloor`	$\uparrow$	`\uparrow`
$)$	`)`	$\lceil$	`\lceil`	$\rceil$	`\rceil`	$\downarrow$	`\downarrow`
$[$	`[`	$\langle$	`\langle`	$\rangle$	`\rangle`	$\updownarrow$	`\updownarrow`
$]$	`]`	$/$	`/`	$\backslash$	`\backslash`	$\Uparrow$	`\Uparrow`
$\{$	`\{`	$\vert$	`|`	$\Vert$	`|`	$\Downarrow$	`\Downarrow`
$\}$	`\}`					$\Updownarrow$	`\Updownarrow`

Beispiel: (Ein Punkt dient als unsichtbarer Begrenzer.)

```
\[ y = \left\{ \begin{array}{rcl} -1 & : & x < 0\\
   0 & : & x = 0\\ 1 & : & x > 0 \end{array} \right. \]
```

$$y = \left\{ \begin{array}{rcl} -1 & : & x < 0\\ 0 & : & x = 0\\ 1 & : & x > 0 \end{array} \right.$$

Binäre Operatoren, Relationen, Pfeile

- Binäre Operatoren:

$\pm$	`\pm`	$\cap$	`\cap`	$\mp$	`\mp`	$\cup$	`\cup`
$\setminus$	`\setminus`	$\uplus$	`\uplus`	$\cdot$	`\cdot`	$\sqcap$	`\sqcap`
$\times$	`\times`	$\sqcup$	`\sqcup`	$\ast$	`\ast`	$\triangleleft$	`\triangleleft`
$\star$	`\star`	$\wr$	`\wr`	$\diamond$	`\diamond`	$\triangleright$	`\triangleright`
$\circ$	`\circ`	$\bigcirc$	`\bigcirc`	$\bullet$	`\bullet`	$\bigtriangleup$	`\bigtriangleup`
$\div$	`\div`	$\oslash$	`\oslash`	$\vee$	`\vee`	$\bigtriangledown$	`\bigtriangledown`
$\wedge$	`\wedge`	$\odot$	`\odot`	$\oplus$	`\oplus`	$\dagger$	`\dagger`
$\ominus$	`\ominus`	$\ddagger$	`\ddagger`	$\otimes$	`\otimes`	$\amalg$	`\amalg`

- Relationen:

$\leq$	`\leq`	$\geq$	`\geq`	$\prec$	`\prec`	$\succ$	`\succ`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\ll$	`\ll`	$\gg$	`\gg`
$\subset$	`\subset`	$\supset$	`\supset`	$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\in$	`\in`	$\ni$	`\ni`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\smile$	`\smile`	$\mid$	`\mid`
$\frown$	`\frown`	$\parallel$	`\parallel`	$=$	`=`	$\cong$	`\cong`
$\equiv$	`\equiv`	$\bowtie$	`\bowtie`	$\sim$	`\sim`	$\propto$	`\propto`
$\simeq$	`\simeq`	$\models$	`\models`	$\asymp$	`\asymp`	$\doteq$	`\doteq`
$\approx$	`\approx`	$\perp$	`\perp`	$<$	`<`	$>$	`>`

- Pfeile:

	$\rightleftharpoons$ `\rightleftharpoons`	
$\leftarrow$ `\leftarrow`	$\longleftarrow$ `\longleftarrow`	$\uparrow$ `\uparrow`
$\Leftarrow$ `\Leftarrow`	$\Longleftarrow$ `\Longleftarrow`	$\Uparrow$ `\Uparrow`
$\rightarrow$ `\rightarrow`	$\longrightarrow$ `\longrightarrow`	$\downarrow$ `\downarrow`
$\Rightarrow$ `\Rightarrow`	$\Longrightarrow$ `\Longrightarrow`	$\Downarrow$ `\Downarrow`
$\leftrightarrow$ `\leftrightarrow`	$\longleftrightarrow$ `\longleft...`	$\updownarrow$ `\updownarrow`
$\Leftrightarrow$ `\Leftrightarrow`	$\Longleftrightarrow$ `\Longleft...`	$\Updownarrow$ `\Updownarrow`
$\mapsto$ `\mapsto`	$\longmapsto$ `\longmapsto`	$\nearrow$ `\nearrow`
$\hookleftarrow$ `\hookleftarrow`	$\hookrightarrow$ `\hookrightarrow`	$\searrow$ `\searrow`
$\leftharpoonup$ `\leftharpoonup`	$\rightharpoonup$ `\rightharpoonup`	$\swarrow$ `\swarrow`
$\leftharpoondown$ `\leftharpoondown`	$\rightharpoondown$ `\rightharpoondown`	$\nwarrow$ `\nwarrow`

Die allermeisten der hier abgebildeten Relationsbefehle können durch voranstellen des Befehls `\not` negiert werden. So ergibt `\not=` z.B. $\neq$ und aus `\not\in` wird $\not\in$.

Mathematische Symbole: Log-artige, Spezial

- log-artige Funktionen/Symbole (log-like functions): Damit LaTeX beim Setzen einer Funktion wie $\log$ auch weiß, daß es sich um eine solche handelt, und nicht etwa um das Produkt der drei Variablen l, o und g, gibt es diese Funktionen/Symbole als eigene Befehle im MM:

\arccos	\cot	\exp	\lg	\log	\sin
\arcsin	\coth	\gcd	\lim	\max	\sinh
\arctan	\csc	\hom	\liminf	\min	\sup
\arg	\deg	\inf	\limsup	\Pr	\tan
\cos	\det	\ker	\ln	\sec	\tanh
\cosh	\dim				

- Verschiedene Spezialsymbole:

$\aleph$	\aleph	$\bot$	\bot	$\hbar$	\hbar	$\spadesuit$	\spadesuit
$\imath$	\imath	$\angle$	\angle	$\jmath$	\jmath	$\diamondsuit$	\diamondsuit
ℓ	\ell	$\top$	\top	$\wp$	\wp	$\forall$	\forall
$\Re$	\Re	$\exists$	\exists	$\Im$	\Im	$\neg$	\neg
∂	\partial	$\flat$	\flat	∞	\infty	$\natural$	\natural
$\prime$	\prime	$\sharp$	\sharp	$\emptyset$	\emptyset	$\clubsuit$	\clubsuit
∇	\nabla	$\triangle$	\triangle	$\surd$	\surd	$\heartsuit$	\heartsuit

- Spezialsymbole im `latexsym`-Paket:

$\Box$ \Box	$\Diamond$ \Diamond	$\mho$ \mho

- Weitere Symbole des *latexsym*-Pakets:

$\lhd$ \lhd	$\rhd$ \rhd	$\unlhd$ \unlhd	$\unrhd$ \unrhd
$\sqsubset$ \sqsubset	$\sqsupset$ \sqsupset	$\Join$ \Join	$\leadsto$ \leadsto

$\mathcal{AMS}$-$\TeX$

- Und dann gibt es z.B. noch packages der $\mathcal{A}$merican-$\mathcal{M}$athematical-$\mathcal{S}$ociety, mit deren Hilfe man hunderte weiterer Symbole zur Verfügung hat. An dieser Stelle nur eine kleine Auswahl:

$\leqq$	`\leqq`	$\leqslant$	`\leqslant`	$\eqslantless$	`\eqslantless`
$\lesssim$	`\lesssim`	$\lessapprox$	`\lessapprox`	$\approxeq$	`\approxeq`
$\lessdot$	`\lessdot`	$\lll$	`\lll`	$\lessgtr$	`\lessgtr`
$\lesseqgtr$	`\lesseqgtr`	$\lesseqqgtr$	`\lesseqqgtr`	$\doteqdot$	`\doteqdot`
$\risingdotseq$	`\risingdotseq`	$\fallingdotseq$	`\fallingdotseq`	$\backsim$	`\backsim`
$\backsimeq$	`\backsimeq`	$\subseteqq$	`\subseteqq`	$\Subset$	`\Subset`
$\sqsubset$	`\sqsubset`	$\preccurlyeq$	`\preccurlyeq`	$\curlyeqprec$	`\curlyeqprec`
$\precsim$	`\precsim`	$\precapprox$	`\precapprox`	$\vartriangleleft$	`\vartriangleleft`
$\pitchfork$	`\pitchfork`	$\vDash$	`\vDash`	$\Vvdash$	`\Vvdash`
$\smallsmile$	`\smallsmile`	$\smallfrown$	`\smallfrown`	$\bumpeq$	`\bumpeq`
$\Bumpeq$	`\Bumpeq`	$\geqq$	`\geqq`	$\geqslant$	`\geqslant`
$\eqslantgtr$	`\eqslantgtr`	$\gtrsim$	`\gtrsim`	$\gtrapprox$	`\gtrapprox`
$\gtrdot$	`\gtrdot`	$\ggg$	`\ggg`	$\gtrless$	`\gtrless`
$\gtreqless$	`\gtreqless`	$\gtreqqless$	`\gtreqqless`	$\eqcirc$	`\eqcirc`
$\circeq$	`\circeq`	$\triangleq$	`\triangleq`	$\thicksim$	`\thicksim`
$\thickapprox$	`\thickapprox`	$\supseteqq$	`\supseteqq`	$\Supset$	`\Supset`
$\sqsupset$	`\sqsupset`	$\succcurlyeq$	`\succcurlyeq`	$\curlyeqsucc$	`\curlyeqsucc`
$\succsim$	`\succsim`	$\succapprox$	`\succapprox`	$\vartriangleright$	`\vartriangleright`
$\shortmid$	`\shortmid`	$\Vdash$	`\Vdash`	$\trianglerighteq$	`\trianglerighteq`
$\shortparallel$	`\shortparallel`	$\between$	`\between`	$\trianglelefteq$	`\trianglelefteq`
$\varpropto$	`\varpropto`	$\therefore$	`\therefore`	$\blacktriangleleft$	`\blacktriangleleft`
$\backepsilon$	`\backepsilon`	$\because$	`\because`	$\blacktriangleright$	`\blacktriangleright`

Mathematische Symbole: Alphabete, Schriften

- Griechische Großbuchstaben:

Γ \Gamma	Λ \Lambda	Σ \Sigma	Ψ \Psi
Δ \Delta	Ξ \Xi	Υ \Upsilon	Ω \Omega
Θ \Theta	Π \Pi	Φ \Phi	

- Griechische Kleinbuchstaben: (mit den im Formelsatz üblichen Varianten):

α \alpha	ζ \zeta	λ \lambda	ϖ \varpi	υ \upsilon
β \beta	η \eta	μ \mu	ρ \rho	ϕ \phi
γ \gamma	θ \theta	ν \nu	ϱ \varrho	φ \varphi
δ \delta	ϑ \vartheta	ξ \xi	σ \sigma	χ \chi
ϵ \epsilon	ι \iota	o o	ς \varsigma	ψ \psi
ε \varepsilon	κ \kappa	π \pi	τ \tau	ω \omega

- Kalligraphische Großbuchstaben:

$\mathcal{COMPUTERKALLI}$ `$\mathcal{COMPUTERKALLI}$`

- Kalligraphische Kleinbuchstaben: Gibt es nicht. Dafür aber:

- Mathematische Zeichensätze:

$\mathit{math\ italic\ \alpha_1}$	`$\mathit{math\ italic\ \alpha_1}$`
$\mathrm{math\ roman\ \beta_2}$	`$\mathrm{math\ roman\ \beta_2}$`
$\mathbf{math\ bold\ \gamma_3}$	`$\mathbf{math\ bold\ \gamma_3}$`
$\mathsf{math\ sans\ serif\ \delta_4}$	`$\mathsf{math\ sans\ serif\ \delta_4}$`
$\mathtt{math\ typewriter\ \epsilon_5}$	`$\mathtt{math\  typewriter\ \epsilon_5}$`

4.2 Konstruktionselemente für den Formelsatz

4.2.1

Justieren von Abständen

- Vordefinierte Abstände in LaTeX:

`\quad`	so breit, wie ein Buchstabe hoch ist
`\qquad`	zwei `\quad`
`\!`	-3/18 eines quad
`\,`	3/18 eines quad
`\:`	4/18 eines quad
`\;`	5/18 eines quad

Beispiele:

vorher	verbesserte Eingabe	nachher
$\sqrt{2}x$	`$\sqrt{2}\,x$`	$\sqrt{2}\,x$
$[0,1)$	`$[\,0,1)$`	$[\,0,1)$
$x^2/2$	`$x^2\!/2$`	$x^2\!/2$
$\Gamma_2+\Delta^2$	`$\Gamma_{\!2}+\Delta^{\!2}$`	$\Gamma_{\!2}+\Delta^{\!2}$
$\int\int_D dxdy$	`$ \int\!\!\!\int_D dx\,dy $`	$\int\!\!\!\int_D dx\,dy$

Schriftgrößen beim Formelsatz

- Vordefinierte Schriften:

`\displaystyle`	abgesetzte Formel
`\textstyle`	Textformel
`\scriptstyle`	1. Umstellung
`\scriptscriptstyle`	2. Umstellung

Beispiel vorher:

```
\[
   a_0 + \frac{1}{a_1 +
         \frac{1}{a_2 +
         \frac{1}{a_3}}}
\]
```

$$a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3}}}$$

Beispiel nachher:

```
\[
   a_0 + \frac{1}{\displaystyle a_1 +
         \frac{1}{\displaystyle a_2 +
         \frac{1}{\displaystyle a_3}}}
\]
```

$$a_0 + \frac{1}{\displaystyle a_1 + \frac{1}{\displaystyle a_2 + \frac{1}{\displaystyle a_3}}}$$

Theorem I

Im mathematischen Textsatz benötigt man in der Regel Strukturelemente wie Theoreme, Axiome, Annahmen, Lemmas etc.. Das `\newtheorem` Kommando in LaTeX deckt im Prinzip mit *einem* Befehl *all* diese Fälle ab, indem es Möglichkeiten bereitstellt, *eigene* „Theorem"-Umgebungen zu definieren:

- Definition:
  `\newtheorem{`*Umgebungsbezeichner*`}{`*Umgebungsname*`}[`*Zähler*`]`

Parameter	Beschreibung
Umgebungsbezeichner	Logischer Bezeichner der neuen Umgebung, also der Name, unter dem sich die neue Umgebung jetzt innerhalb der Eingabedatei ansprechen läßt.
Umgebungsname	„Wirklicher" Umgebungsname, wie er später von LaTeX gesetzt werden soll.
Zähler	Zähler für die neue Umgebung; wird bei Umgebungsaufruf weitergezählt. Soll der Zähler von einem zweiten Zähler abhängen (etwa, um Sätze innerhalb jedes Kapitels neu zu numerieren), läßt sich letzterer unter dem optionalen Argument *Zähler* angeben.

- Aufruf:
  `\begin{`*Umgebungsbezeichner*`}[`*Zusatz*`]`
  Text
  `\end{`*Umgebungsbezeichner*`}`

Parameter	Beschreibung
Umgebungsbezeichner	siehe oben
Zusatz	Ein Zusatz wie zum Beispiel der Name der Person, auf die das „Theorem" zurückgeführt wird.

Theorem II

Beispiel:

```
\newtheorem{satzumgebung}{Satz}[section]
\begin{satzumgebung}[Bolzano] Jede beschr"ankte unendliche
  Punktmenge besitzt mindestens einen H"aufungspunkt.
\end{satzumgebung}
```

Satz 4.2.1 (Bolzano) *Jede beschränkte unendliche Punktmenge besitzt mindestens einen Häufungspunkt.*

Gelegentlich kommt es vor, daß mehrere Theorem-Umgebungen einen gemeinsamen Zähler nutzen sollen. Dieser Fall erfordert eine Sonderform der Definition:

- Definition mit gleichem Zähler:
  `\newtheorem{`*Umgebung_2*`}[`*Umgebung_1*`]{`*Name_2*`}`

Parameter	Beschreibung
Umgebung_2	Der Umgebungsbezeichner einer zweiten Umgebung.
Umgebung_1	Der Umgebungsbezeichner der bereits definierten Umgebung, deren Zähler von *Umgebung_2* mitbenutzt werden soll.
Umgebungsname_2	Der Umgebungsname der hier definierten zweiten Umgebung, wie er von LaTeX gesetzt werden soll.

Beispiel zu gemeinsam nutzbaren Zählern

Folgende Umgebungen werden vom Beispiel vorausgesetzt:

```
\newtheorem{theo}{Satz}[section]
\newtheorem{Def}[theo]{Definition}
```

```
\begin{Def}
  Sei $f$ eine ${\cal C}^3$-Abbildung.\\
  Die {\bf Schwarz'sche Ableitung} ist definiert durch
  \[
    Sf(x) := \frac{f'''(x)}{f'(x)} - \frac{3}{2}
    \left( \frac{f''(x)}{f'(x)} \right)^2,
    \quad\mbox{ f"ur } x \ne 0.
  \]
\end{Def}
```

Definition 4.2.1 *Sei f eine $\mathcal{C}^3$-Abbildung.*
Die **Schwarz'sche Ableitung** *ist definiert durch*

$$Sf(x) := \frac{f'''(x)}{f'(x)} - \frac{3}{2}\left(\frac{f''(x)}{f'(x)}\right)^2, \quad \text{für } x \neq 0.$$

```
\begin{theo}[cf.~Choe\cite{Choe}]\label{ergo}
  Sei $f:[-1,1]\to[-1,1]$ {$\cal A$}-me"sbar und
  das Ma"s $\nu$ f-ergodisch. Dann gilt
  f"ur alle $\;h\in L^1\left( [-1,1],\nu \right)$
  \begin{equation}
    \lim_{N\to\infty} \frac{1}{N}\sum_{i=0}^{N-1}
    h\left(f^i(x)\right) = \int h\, d\nu
  \end{equation}
  f"ur $x\in[-1,1] \quad\nu$-fast "uberall.
\end{theo}

Satz \ref{ergo} ist das Ergodentheorem ...
```

Satz 4.2.2 (cf. Choe [C1]) *Sei $f : [-1,1] \to [-1,1]$ $\mathcal{A}$-meßbar und das Maß ν f-ergodisch. Dann gilt für alle $h \in L^1([-1,1],\nu)$*

$$\lim_{N\to\infty} \frac{1}{N}\sum_{i=0}^{N-1} h\left(f^i(x)\right) = \int h\, d\nu \tag{4.5}$$

für $x \in [-1,1]$ ν-fast überall.

Satz 4.2.2 ist das Ergodentheorem ...

picture und graphics im MM

Weigert sich LaTeX hartnäckig, eine Formel wie gewünscht zu setzen, kann man noch die *picture*-Umgebung und das *graphics*-package zu Hilfe nehmen.

- Die *picture*-Umgebung liefert die Möglichkeit, Boxen jedweden Inhalts (z.B. Text, Punkte, Linienelemente) quasi beliebig zu positionieren.

  Beispiel: Definition eines `\widevec`-Befehls.

```
\newlength{\vwdth} \newlength{\vhght}
\newcommand{\widevec}[1]%
{\settowidth{\vwdth}{#1}\settoheight{\vhght}{#1}%
  ⋮  (Arithmetik, siehe Ergänzungsseite)
 \setlength{\unitlength}{\vhght}%
 \begin{picture}(\bwdth,\bhght)
 \put(0,0){\makebox(0,0)[bl]{#1}}
 \put(0,\bhght){\vector(1,0){\bwdth}}
 \end{picture}}
Dieser \widevec{Vektor} ist ein "‘Weitvektor"’.
```

  Dieser $\overrightarrow{\text{Vektor}}$ ist ein „Weitvektor“.

- Das *graphics*-package ermöglicht es, Boxen zu rotieren.

  Beispiel: Einführung von „Schrägvektoren“.

```
\parbox[c]{2cm}{\rotatebox{45}{%
$\left(\begin{array}{c}a\\b\\c\end{array}\right)$}}
{bildet ein Produkt mit}
\parbox[c]{2cm}{\rotatebox{-45}{%
$\left(\begin{array}{c}d\\e\\f\end{array}\right)$}}
{wie folgt:}
```

  $\left(\begin{array}{c}a\\b\\c\end{array}\right)$ bildet ein Produkt mit $\left(\begin{array}{c}d\\e\\f\end{array}\right)$ wie folgt:

Erläuterung des Beispiels

Unten findet sich das Beispiel der Vorseite *vollständig* wiedergegeben. Drei von den vier vorher verborgenen Zeilen enthalten plain-TeX-Befehle, ohne die es bei der Definition eigener Befehle oft nicht geht.

In Zeile 1 werden zwei Abstandsregister angelegt, genannt `\vwdth` und `\vhght`, zur Aufnahme von Breite und Höhe der Box, über die der Vektorpfeil zu zeichnen ist (Eingangsbox). Die Zuweisung dieser beiden Werte erfolgt in Zeile 3, also innerhalb der Deklaration des neuen Befehls, die in Zeile 2 eingeleitet wird. In Zeile 4 werden zwei weitere Register angelegt, diesmal jedoch *Ganzzahlregister* von TeX. Sie dienen zur Aufnahme von Breite und Höhe der Box *inklusive* Vektorpfeil (Ausgangsbox), welche in Zeile 5 zugewiesen werden, aber Achtung:

Das Problem dieses Beispiels liegt in der Arithmetik von LaTeX bzw. TeX. In der *picture*-Umgebung wird nicht mit Abständen, sondern mit Vielfachen des Abstands `\unitlength` gerechnet. Um solche Vielfache angeben zu können, muß man sich der Ganzzahlarithmetik von TeX bedienen.

Hierzu werden im Beispiel die Positionen in Vielfachen von einem tausendstel der Eingangsboxhöhe ausgedrückt. Die y-Position des zu zeichnenden Vektorpfeils soll jetzt 1.2 mal die Höhe der Eingangsbox sein, deshalb wird `\bhght` auf 1200 gesetzt. Die Länge des Vektorpfeils soll so groß sein wie die Breite der Eingangsbox — in Einheiten eines tausendstels der Eingangsboxhöhe! Wie *rechnet* man das?

Der Befehl `\bwdth=\number\vwdth` weist dem Register `\bwdth` den Wert von `\vwdth` in Ganzzahleinheiten zu, wie sie allen Berechnungen von TeX zugrundeliegen. In Zeile 7 wird nun eine Ganzzahldivision durchgeführt, und zwar wird die soeben ermittelte Zahl durch ein tausendstel der Eingangsboxhöhe (ausgedrückt in Ganzzahleinheiten) geteilt. Das Ergebnis ist die gesuchte Breite der Eingangsbox in tausendstel Eingangsboxhöhen (vom Rundungsfehler der Ganzzahldivision abgesehen).

Damit liegen alle benötigten Größen vor, und die *picture*-Umgebung kann — mit der richtigen `\unitlength` — aufgerufen werden (Zeilen 8 und 9). Höhe und Breite des Bildes sind Höhe und Breite der Ausgangsbox!

Im Bild wird zunächst die Eingangsbox ausgegeben (was immer Argument 1 auch enthielt, Zeile 10) und danach ein Vektorpfeil der Länge `\bwdth` von Position $x = 0$, $y =$`\bhght` flach nach rechts (Zeile 11).

```
 1 \newlength{\vwdth} \newlength{\vhght}
 2 \newcommand{\widevec}[1]%
 3 {\settowidth{\vwdth}{#1}\settoheight{\vhght}{#1}%
 4  \countdef\bwdth=1 \countdef\bhght=2%
 5  \bwdth=\number\vwdth \bhght=1200%
 6  \setlength{\vhght}{.001\vhght}%
 7  \divide\bwdth by \number\vhght%
 8  \setlength{\unitlength}{\vhght}%
 9  \begin{picture}(\bwdth,\bhght)
10  \put(0,0){\makebox(0,0)[bl]{#1}}
11  \put(0,\bhght){\vector(1,0){\bwdth}}
12  \end{picture}}
```

Stilparameter

Folgende Parameter beeinflussen das Layout im mathematischen Modus und sollten bis auf `jot` elastische Maßangaben enthalten:

Parameter	Beschreibung
`\arraycolsep`	Halbe Breite des Spaltenzwischenraums.
`\arraystretch`	Dehnfaktor des Zeilenzwischenraums (zu benutzen mit `\renewcommand` !).
`\jot`	Zusätzlicher vertikaler Zwischenraum bei `eqnarray(*)`.
`\mathindent`	Einrücktiefe links bei Verwendung der Dokumentklassenoption `fleqn`.
`\abovedisplayskip`	Zusätzlicher vertikaler Zwischenraum über langen Formeln.
`\belowdisplayskip`	Zusätzlicher vertikaler Zwischenraum unter langen Formeln.
`\abovedisplayshortskip`	Zusätzlicher vertikaler Zwischenraum über kurzen Formeln.
`\belowdisplayshortskip`	Zusätzlicher vertikaler Zwischenraum unter kurzen Formeln.
`\topsep`	Wird bei Verwendung der Dokumentklassenoption `fleqn` anstelle der vier letztgenannten Parameter benutzt.

4.3 Das LaTeX-Fontsystem

4.3.1

NFSS

Nicht alle in LaTeX prinzipiell verfügbaren Fonts lassen sich mit den sog. *High Level* Befehlen `\textsf{...}`, `\ttfamily{...}`, ... ansprechen.

Um auch *externe* Fonts (z.B. Adobe) ansprechen zu können, braucht man eine genaue Fontklassifikation.

Die Fonts werden nach folgenden Kriterien klassifiziert:

- Die Kodierung des Fonts, d.h. wie ein Zeichen intern von LaTeX angesprochen werden kann.

  Die Computer Modern Schriftfamilie benutzt die sog. `OT1` Kodierung.
- Die Schriftfamilie. Zum Beispiel `cmr` für die Computer Modern Roman Fonts oder `panr` für die Pandora Roman Fonts.
- Der Schriftschnitt. Zum Beispiel `m` für Buchschrift.
- Die Schriftauszeichnung des Fonts. Zum Beispiel `sl` für eine schräge Schrift.

Die Kodierung der Schriftfamilie und der Auszeichnung stimmt mit den High Level Befehlen überein.

- Die Fontklassifikation, die Befehle zur Fontauswahl und die Befehle zur Bereitstellung neuer Fonts bilden zusammen das *NFSS*.
- Diese Abkürzung steht für *New Font Selection Scheme*.
- Die „alten“ Fontauswahlbefehle von LaTeX 2.09 (wie z.B `\it`) sind mit den NFSS-Befehlen nicht verträglich.
- Mit dem Paket *oldlfont* können die alten LaTeX 2.09 Befehle zur Fontauswahl weiter verwendet werden.
- Einige mathematische Symbole (z.B. `\Box`), die unter LaTeX 2.09 zur Verfügung standen, sind in der jetzigen Version von LaTeX nicht mehr vorhanden. Mit dem Paket *latexsym* können diese Symbole weiter verwendet werden.

NFSS: Kodierung

Die Standardkodierung von LaTeX heißt OT1. Die Zeichen eines Fonts sind durchnumeriert und lassen sich über ihre jeweilige Nummer ansprechen.

Abbildung 4.1 zeigt die 128 Zeichen des Fonts cmr10 als OT1-Kodierung in Form einer kombinierten Oktal/Hexadezimal-Tabelle.

	´0	´1	´2	´3	´4	´5	´6	´7	
´00x	Γ	Δ	Θ	Λ	Ξ	Π	Σ	ϒ	″0x
´01x	Φ	Ψ	Ω	ﬀ	ﬁ	ﬂ	ﬃ	ﬄ	
´02x	ı	ȷ	`	´	ˇ	˘	¯	˚	″1x
´03x	¸	ß	æ	œ	ø	Æ	Œ	Ø	
´04x	-	!	”	#	$	%	&	’	″2x
´05x	(	)	*	+	,	-	.	/	
´06x	0	1	2	3	4	5	6	7	″3x
´07x	8	9	:	;	¡	=	¿	?	
´10x	@	A	B	C	D	E	F	G	″4x
´11x	H	I	J	K	L	M	N	O	
´12x	P	Q	R	S	T	U	V	W	″5x
´13x	X	Y	Z	[	“	]	ˆ	˙	
´14x	‘	a	b	c	d	e	f	g	″6x
´15x	h	i	j	k	l	m	n	o	
´16x	p	q	r	s	t	u	v	w	″7x
´17x	x	y	z	–	—	˝	˜	¨	
	″8	″9	″A	″B	″C	″D	″E	″F	

Abbildung 4.1: OT1-Kodierung von cmr10

Im Jahr 1990 ist eine neue Kodierung für LaTeX entworfen worden, die sog. *Cork Encoding*. Diese Kodierung wird mit T1 bezeichnet und kann 256 Zeichen erfassen, also doppelt so viele, wie die OT1 Kodierung.

- Dies ist von Vorteil, wenn Umlaute und Buchstaben mit Akzenten verwendet werden müssen, da diese Zeichen in der neuen Kodierung trennbar sind.
- Zu der Kodierung T1 gehören eigene Fonts, die DC Fonts.
- Diese Fonts können mit dem Paket *t1enc* geladen werden und ersetzen dann die Computer Modern Fonts.

NFSS: Schriftschnitt und Auszeichnung

Das NFSS kennt die folgenden Spezifikationen für den Schriftschnitt; dabei wird nochmals das „Gewicht“ des Fonts und seine „Weite“ unterschieden.

- Das Gewicht eines Fonts ist die „Dicke“ der Linien, aus denen die Buchstaben gezeichnet sind.
- Die Weite eines Fonts gibt seine horizontale Streckung an.

Gewicht		Weite		
Symbol	engl. Name	Symbol	engl. Name	%
`ul`	Ultra Light	`uc`	Ultra Condensed	50.0
`el`	Extra Light	`ec`	Extra Condensed	62.5
`l`	Light	`c`	Condensed	75.0
`sl`	Semi Light	`sc`	Semi Condensed	87.5
`m`	Medium	`m`	Medium	100.0
`sb`	Semi Bold	`se`	Semi Expanded	112.5
`b`	Bold	`x`	Expanded	125.0
`eb`	Extra Bold	`ex`	Extra Expanded	150.0
`ub`	Ultra Bold	`ux`	Ultra Expanded	200.0

- Wird nur das Gewicht angegeben, wird eine normale Weite (= Medium) angenommen
- Soll ein Gewicht in seiner Weite geändert werden, wird das entsprechende Weiten-Symbol einfach angehängt. Zum Beispiel bezeichnet `sbc` einen Semi Bold Condensed Font.
- Es stehen nicht alle Schnittkombinationen in allen Fonts zur Verfügung.

NFSS kennt folgende Spezifikationen zur Auszeichnung:

Symbol	Bedeutung
`n`	aufrechte (normale) Schrift
`it`	Kursive Schrift
`sl`	schräge Schrift
`sc`	Kapitälchen
`ui`	aufrechte Kursivschrift

Die Fontklassifikation wird in Handbüchern zuweilen in der Form

`OT1/cmr/m/n/10` oder `T1/dcb/m/sl/14.4`

angegeben. Die Fonts können so *nicht* geladen werden!

NFSS: Laden neuer Fonts

- Mit dem \usefont Befehl kann ein Font direkt über seine NFSS Klassifikation angesprochen werden:

  \usefont{*cod*}{*fam*}{*schnitt*}{*ausz*}

  (*cod* = Fontkodierung, *fam* = Fontfamilie, *schnitt* = Schriftschnitt, *ausz* = Auszeichnung des Fonts.)

- Der \usefont Befehl muß mit \selectfont abgeschlossen werden.

  Beispiel: \usefont{OT1}{cmdh}{m}{n}\selectfont

- Es lassen sich eigene Schriftfamilienbefehle definieren:

```
\newcommand{\dhfamily}{\usefont{OT1}{cmdh}{m}{n}%
     \selectfont}
\dhfamily Dies ist in Dunhill geschrieben.
```

Dies ist in Dunhill geschrieben.

- Die voreingestellten Fonts lassen sich mit folgenden Parametern ändern:

Parameter	Voreinstellung	Benutzt für
\encodingdefault	OT1	Kodierung
\familydefault	\rmdefault	Familie der Normalschrift
\shapedefault	m	Schnitt der Normalschrift
\seriesdefault	n	Auszeichnung der Normalschrift
\rmdefault	cmr	\rmfamily
\sfdefault	cmss	\sffamily
\ttdefault	cmtt	\ttfamily
\bfdefault	bx	\bfseries
\mddefault	m	\mdseries
\itdefault	it	\itshape
\sldefault	sl	\slshape
\scdefault	sc	\scshape
\updefault	n	\upshape

Beispiel: \renewcommand{\rmdefault}{panr}

NFSS Klassifizierung der Computer Modern Fonts

Die folgende Tabelle gibt die Klassifikation der Computer Modern Schriftfamilien wieder. Mit dem Befehl

```
\usefont{OT1}{cmdh}{m}{n}\selectfont
```

kann dann z.B. der Dunhill Font benutzt werden.

Roman				
Kodierung	Familie	Schnitt	Auszeichnung	Beispiel
OT1	cmr	m	n	Computer Modern
OT1	cmr	m	it	*Computer Modern*
OT1	cmr	m	sl	*Computer Modern*
OT1	cmr	m	sc	COMPUTER MODERN
OT1	cmr	bx	n	**Computer Modern**
OT1	cmr	bx	it	***Computer Modern***
OT1	cmr	bx	sl	***Computer Modern***
OT1	cmr	b	n	**Computer Modern**

Sans serif				
Kodierung	Familie	Schnitt	Auszeichnung	Beispiel
OT1	cmss	m	n	Computer Modern
OT1	cmss	m	sl	*Computer Modern*
OT1	cmss	bx	n	**Computer Modern**
OT1	cmss	sbc	n	**Computer Modern**

Schreibmaschine				
Kodierung	Familie	Schnitt	Auszeichnung	Beispiel
OT1	cmtt	m	n	`Computer Modern`
OT1	cmtt	m	it	*Computer Modern*
OT1	cmtt	m	sl	*Computer Modern*
OT1	cmtt	m	sc	COMPUTER MODERN

Fibonacci				
Kodierung	Familie	Schnitt	Auszeichnung	Beispiel
OT1	cmfib	m	n	Computer Modern

Funny Roman				
Kodierung	Familie	Schnitt	Auszeichnung	Beispiel
OT1	cmfr	m	n	Computer Modern
OT1	cmfr	m	it	*Computer Modern*

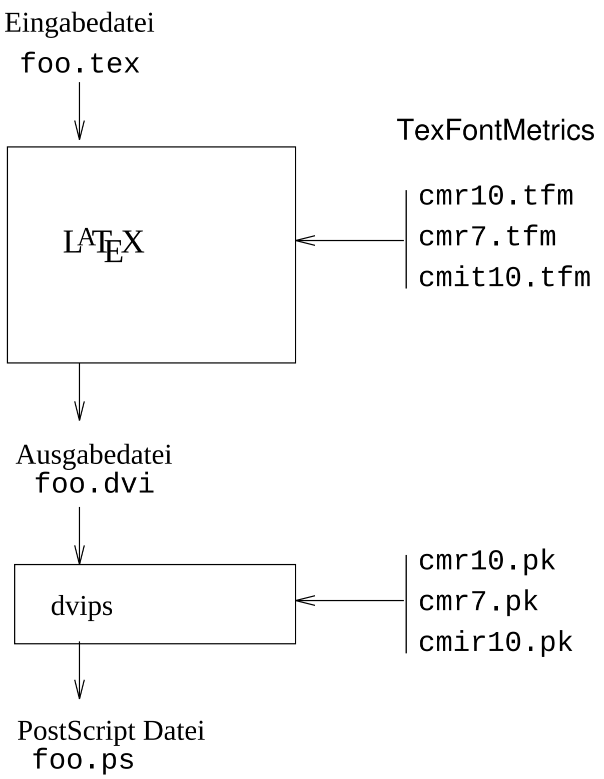
Dunhill				
Kodierung	Familie	Schnitt	Auszeichnung	Beispiel
OT1	cmdh	m	n	Computer Modern

Wie arbeitet LaTeX mit Fonts?

LaTeX verwendet für die Formatierung des Textes nur Informationen über die Größe der Buchstaben (genauer: der umgebenden Boxen). Diese Informationen sind in den `.tfm` Dateien abgelegt.

- Die `.tfm` Dateien existieren für jeden Font und sind in binärer Form abgelegt. Für die Normalschrift (roman) verwendet LaTeX z.B. `cmr10.tfm` als `.tfm` Datei.

- Für den Ausdruck oder die Ansicht im Previewer existieren für jeden Font Pixelgraphiken der einzelnen Buchstaben. Diese sind in den `.pk`-Dateien abgelegt. Für die Normalschrift in 10pt Entwurfgröße bei einer Auflösung von 300 dpi[3] existiert z.B. die `.pk`-Datei `cmr10.300pk`.



- LaTeX ersetzt den Text in `foo.tex` durch die Größeninformationen aus den `.tfm` Dateien der benutzten Fonts und formatiert die Seiten.

- Der Text und die Positionen der Buchstaben auf der Seite werden in die Datei `foo.dvi` geschrieben.

- Der Druckertreiber (z.B. dvips) ersetzt die Buchstaben aus `foo.dvi` durch die Pixelgraphik der Fonts (`.pk` Dateien).

[3] „dots per inch“; je höher die Auflösung, desto höher die erreichbare Druckgüte.

METAFONT

Das Programm METAFONT dient zur Erzeugung von TEX- bzw. LATEX-Zeichensätzen.

- Der Autor von METAFONT ist Donald E. Knuth, der auch das TEX-Programmsystem entwickelt hat.

- METAFONT ist freie Software (also kostenlos).

- Jeder LATEX-Font entsteht aus einer METAFONT Quelldatei (Endung `.mf`), in der die Form der Buchstaben abstrakt beschrieben wird.

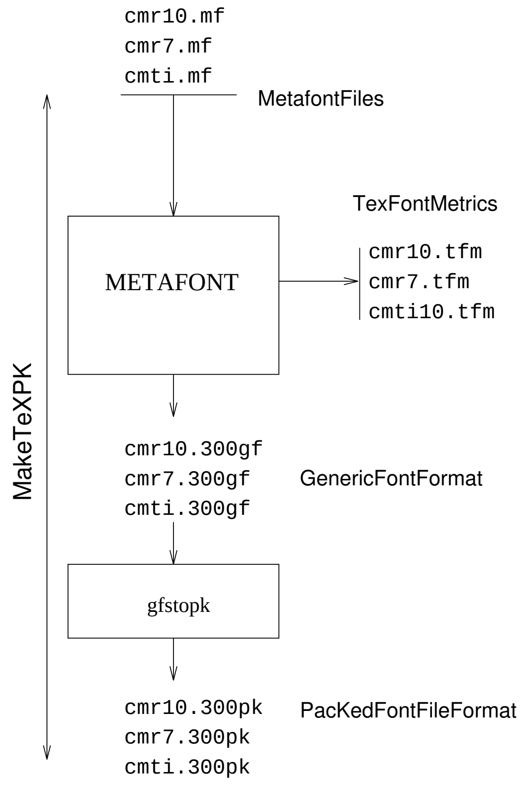
  Beispiel: Die Schrift `OT1/cmr/m/n/10` ist in der Datei `cmr10.mf` definiert:

```
% Computer Modern Roman 10 point
if unknown cmbase: input cmbase fi

font_identifier:="CMR"; font_size 10pt#;

u#:=20/36pt#;          % unit width
width_adj#:=0pt#;      % width adjustment for certain characters
serif_fit#:=0pt#;      % extra sidebar near lowercase serifs
cap_serif_fit#:=5/36pt#;  % extra sidebar near uppercase serifs
letter_fit#:=0pt#;     % extra space added to all sidebars

      :
      :

serifs:=true;          % should serifs and bulbs be attached?
monospace:=false;      % should all characters have the same width?
variant_g:=false;      % should an italic-style g be used?
low_asterisk:=false;   % should the asterisk be centered at the axis?
math_fitting:=false;   % should math-mode spacing be used?

generate roman         % switch to the driver file
```

- METAFONT berechnet aus den `.mf`-Dateien einerseits die Pixelgraphiken der Buchstaben (die später zu `.pk`-Dateien verarbeitet werden) und andererseits die Größe der Buchstaben und deren Abstände untereinander. Diese Informationen schreibt es in die zugehörige `.tfm`-Datei.

METAFONT intern

Die METAFONT Quelldateien (`.mf`-Dateien) enthalten eine abstrakte Beschreibung der Schriften von TeX und LaTeX. Aus dieser Beschreibung erzeugt METAFONT mit Hilfe mathematischer Funktionen (Splines) Pixelgraphiken der Buchstaben. Diese Pixelgraphiken werden passend für das jeweilige Ausgabegerät erzeugt. Daher unterscheiden sich diese Dateien hinsichtlich der Auflösung (also auch: Größe der Datei) für verschiedene Druckertypen und Bildschirme. Das METAFONT Programm berücksichtigt insbesondere die Eigenarten der verschiedenen Druckwerke (Schwärzungsgrad, Art der Linienbildung) bei der Erstellung der Pixelgraphiken. Diese Gerätebeschreibung wird in gesonderten Dateien als `mode` festgehalten.



Die Pixelgraphiken der Buchstaben werden in Dateien mit der Endung `.gf` abgelegt. Neben den `.gf`-Dateien erzeugt METAFONT die `.tfm`-Dateien (TeX Font Metrics), in denen die Abstände zwischen den Buchstaben (Boxen) für den Satz durch TeX (Compile) festgehalten werden. Beispiel eines METAFONT Aufrufs:

```
mf \mode:=CanonCX; mag:=1+0/300;/
          batchmode; input cmr10
```

(Aus `cmr10.mf` errechnet METAFONT die Dateien `cmr10.300gf`. und `cmr10.tfm`)

Die `.gf`-Dateien können zum Druck verwendet werden; sie sind allerdings sehr groß. Daher werden die `.gf`-Dateien mit dem Programm gftopk zu `.pk`-Dateien komprimiert (gepackt), und in dieser Form abgespeichert.

Beispiel: Aus `cmr10.300gf` erzeugt der Programm gftopk die Datei `cmr10.300pk`.

Bei TeX/LaTeX Installationen auf UNIX Betriebssystemen werden die oben beschriebenen Arbeitsgänge durch das Script MakeTeXPK automatisch übernommen. Benutzt man einen noch nicht als `.pk`-Datei vorhandenen Font (z.B. mit `\usefont`) wird vor dem Ausdruck das MakeTeXPK Script gestartet. Die `.mf`-Dateien müssen natürlich vorhanden und für das Script auffindbar sein.

Für die DOS Version von LaTeX (emTeX) sind die `.pk`-Dateien in sog. *Font-Libraries* zusammengefaßt, die das Laden der Fonts für den Previewer und den Drucker erheblich beschleunigen. Sollen zu einem emTeX System weitere (neue) Fonts hinzugefügt werden, so empfiehlt sich ein Studium der Datei

`\EMTEX\DOC\GERMAN\DVIDRV.DVI`

sowie des Programms `FONTLIB`, mit dem Fontlibraries selbst gepackt werden können.

Eine gute Übersicht über das METAFONT Programm erhält man in [7] und [9].

Abschnitt 5

Bibliographie, Graphik und visuelle Formatierung

Lange und anspruchsvolle Texte lassen sich erst über begleitende Arbeitshilfen wie Bibliographie und Indizierung erschließen. Auch werden oft Abbildungen in den Text einbezogen, die helfen sollen, das Geschriebene zu veranschaulichen. LaTeX verfügt über eine Vielzahl von Möglichkeiten, um einen Text um dergleichen Material zu ergänzen.

Da für den Autor nicht nur ein gut lesbares Ergebnis seiner Arbeit wichtig ist, sondern auch, daß diese Arbeit — an LaTeX-Quelldateien — während der Erstellung seines Dokuments überschaubar bleibt, werden abschließend Techniken vorgestellt, mit denen selbst sehr große Dokumente bearbeitbar bleiben.

5.1 Bibliographie und Index

5.1.1

Bibliographie

Ein Literaturverzeichnis wird mit der *thebibliography* Umgebung angelegt.

\begin{thebibliography}{*muster_marke*}
\bibitem[*marke_1*]{*bezug_1*}
eintrag_1
⋮
\end{thebibliography}

- Der `\bibtitem` Befehl versieht die Einträge der Bibliographie mit Marken der Form [1], [2], ... Wird das optionale Argument benutzt, können diese Marken für jeden Eintrag frei bestimmt werden.

- *muster_marke* ist ein Platzhalter für die Breite der Bibliographiemarke, die von `\bibitem` gesetzt wird.

  Beispiele:

1 – 9 Einträge:	*muster_marke* = 9 (für Marken der Form [5])
10 – 99 Einträge:	*muster_marke* = 65
Autorenkürzel:	*muster_marke* = XYZ (für Marken der Form [Kop])

- *bezug_1* bezeichnet eine Bezugs-Marke, auf die im laufenden Text mit

  \cite[*zusatz*]{*bezug*}

  verwiesen werden kann. Optional kann mit *zusatz* weiterer Text nach der Markierung eingefügt werden.

  Beispiel:

```
F"ur eine genaue Beschreibung siehe \cite{lam941} und
\cite[S.~60 ff]{kop941}.
```

Für eine genaue Beschreibung siehe [12] und [10, S. 60 ff].

Beispiel einer Bibliographie

Eine Bibliographie verzeichnet die im Dokument zitierte Literatur. Die Bibliographie kann mit der *thebibliography* Umgebung formatiert werden und wird an deren Stelle ausgegeben.

Beispiel:

```
\begin{thebibliography}{XXX}
\bibitem[Lam]{Lam94} L.~Lamport: {\itshape \LaTeX,
  A Document Preparation System, User's Guide and Reference
  Manual}, 2nd ed, Addison-Wesley Publishing Company (1994)
\bibitem[Kop]{Kop94} H.~Kopka {\itshape \LaTeX, Bd.\ 1.\
  Einf"uhrung}, Addison-Wesley Publishing Company (1994)
\end{thebibliography}
```

wird an dieser Stelle wie folgt ausgedruckt:

Literaturverzeichnis

[Lam] L. Lamport: *LaTeX, A Document Preparation System, User's Guide and Reference Manual*, 2nd ed, Addison-Wesley Publishing Company (1994)

[Kop] H. Kopka *LaTeX, Bd. 1. Einführung*, Addison-Wesley Publishing Company (1994)

Vorgehen von LaTeX:

- LaTeX schreibt die mit `\cite` aufgerufenen Bezugs-Marken in die `.aux` Datei. In dieser steht dann in unserem Beispiel:

  ```
  \citation{Lam94}
  \citation{Kop94}
  ```

- Beim nächsten LaTeX-Lauf werden diese Bezugs-Marken durch die Bibliographie-Marken ersetzt.

- Für die Erstellung einer mit den korrekten Marken gesetzten Bibliographie sind also mindestens zwei LaTeX-Läufe nötig.

BibTEX

Mit dem BIBTEX-Programm kann eine *thebibliography*-Umgebung mit den zitierten Einträgen automatisch erstellt werden.

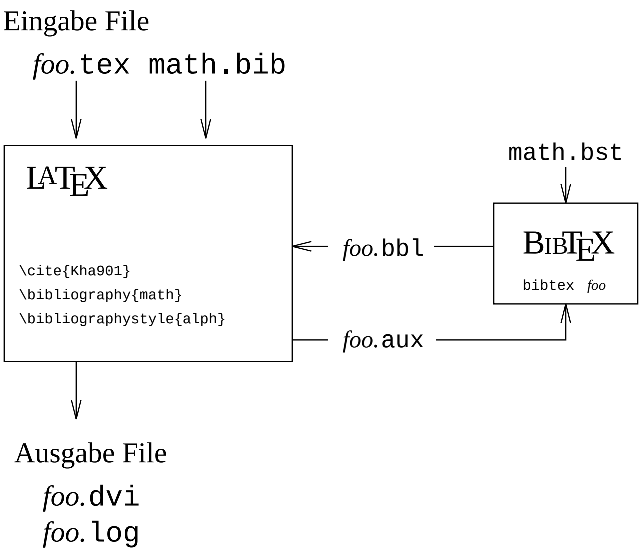
- BIBTEX verwendet dazu eine Datenbank, in der die Bibliographieeinträge in fest definierte Felder eingetragen werden müssen. Diese Datenbank muß in einer Datei mit der Endung `.bib` abgelegt werden, z.B. `math.bib` oder `bio.bib`

- In `foo.tex` ist die `.bib` Datei mit dem Befehl `\bibliography{...}` bekanntzugeben:

  Die Bibliographie erscheint dann an der Stelle dieses Befehls.

```
\documentclass{article}
\begin{document}
          :
\bibliography{math,bio}
\end{document}
```

- BIBTEX liest aus der Datei `foo.aux` die Namen der ausgewählten Datenbanken aus, z.B. `math.bib` und `bio.bib`.



- BIBTEX untersucht die Datenbanken nach Einträgen, die mit `\cite` angewählt wurden. Findet BIBTEX solche Einträge, schreibt es diese in eine Datei namens `foo.bbl`

- Diese Einträge werden in der Form eines `\bibitems` formatiert. BIBTEX verwendet hierzu ein Stylefile mit der Endung `.bst`

- Fehlermeldungen werden in der Datei `foo.blg` festgehalten.

- Es sind ein LATEX, ein BIBTEX und nochmals zwei LATEX-Läufe nötig, um eine korrekt gesetzte Bibliographie zu erhalten.

- Nicht mit `\cite` aufgerufene Einträge können mit `\nocite{*}` hinzugefügt werden.

BibTEX: Ein Beispiel einer Datenbank

Eine BIBTEX Datenbank ist eine Datei mit mit festen Feldeinträgen:

```
@string{lhbook = "Chaotic behavior of
     deterministic systems (LES HOUCHES 1981)"}
@inproceedings{lh:mis,
     AUTHOR = "Misiurewicz, M.",
     TITLE = "Maps of an Interval",
     BOOKTITLE = lhbook,
     PAGES = "565  --  590",
     PUBLISHER = "North-Holland",
     ADDRESS = "Amsterdam -- New York",
     YEAR = "1983"      }
@book{abr&rob,
     AUTHOR = "Abraham, R. and Robbin, J.",
     TITLE = "Transversal Mappings and Flows",
     PUBLISHER = "Benjamin Inc.",
     ADDRESS = "New York",
     YEAR = "1967"      }
@article{cel,
     AUTHOR = "Collet, P. and Eckmann, {J.-P}.
               and Lanford, O. E.",
     TITLE = "Universal Properties of Maps...",
     JOURNAL = "Comm. Math. Phys.",
     PAGES = "211 -- 254",
     YEAR = "1980"      }
```

- Nach dem Zeichen `@` folgt der Eingabetyp: `book`, `article`, ...
- Zu jedem Eingabetyp gehören notwendige und optionale Felder (`AUTHOR`, `TITLE`, . . .).
- Mit dem Befehl `\bibliographystyle{`*stil*`}` können verschiedene Bibliographiestile in `foo.tex` ausgewählt werden:

stil	Bibliographie wird wie folgt gesetzt:
`plain`	Alphabetisch nach Autoren sortiert; Form: [1], [2].
`unsrt`	In der Reihenfolge der `\cite` Befehle.
`alpha`	Alphabetisch nach Autoren sortiert; Form: [Kop], [Lam].
`abbrv`	Wie alpha, jedoch mit automatischer Markengenerierung.

Darüber hinaus existieren viele weitere Bibliographiestile.

BibTEX-Eingabetypen

Das Layout eines Bibliographieeintrags hängt von der Gattung der aufgenommenen Literaturstelle ab: Bücher werden anders zitiert als Preprints. Diesem Umstand trägt BIBTEX durch sog. Eingabetypen (z.B. `BOOK`, `UNPUBLISHED`) Rechnung. Zu jedem Eingabetyp gehören typische Felder (wie `AUTHOR` oder `TITLE`). Man unterscheidet daher für jeden Eingabetyp notwendige und optionale Felder. Bei den Feldeinträgen ist zu beachten, daß die Anführungszeichen als Feldbegrenzer dienen. Umlaute müssen daher in der Form `AUTHOR = "H{"o}pf, P."`, eingegeben werden. Mit solcher Klammerung kann auch eine Großschreibung von Namen und Titeln erzwungen werden. Es folgt eine Auflistung der Eingabetypen mit ihren jeweils notwendigen und optionalen Feldern.

Eingabetyp	notw. Felder	optionale Felder
`article`	`author`, `title`, `journal`, `year`	`volume`, `number`, `pages`, `month`, `note`
`book`	`author` oder `editor`, `title`, `publisher`, `year`	`volume` oder `number`, `series`, `address`, `edition`, `month`, `note`
`booklet`	`title`	`author`, `howpublished`, `address`, `month`, `year`, `note`
`inbook`	`author` oder `editor`, `title`, `chapter` oder `pages`, `publisher`, `year`	`volume` oder `number`, `series`, `type`, `address`, `edition`, `month`, `note`
`incollection`	`author`, `title`, `booktitle`, `publisher`, `year`	`editor`, `volume` oder `number`, `series`, `type`, `chapter`, `pages`, `address`, `edition`, `month`, `note`
`inproceedings`	`author`, `title`, `booktitle`, `year`	`editor`, `volume` oder `number`, `series`, `pages`, `address`, `month`, `organization`, `publisher`, `note`
`manual`	`title`	`author`, `organization`, `address`, `edition`, `month`, `year`, `note`
`masterthesis`	`author`, `title`, `school`, `year`	`type`, `address`, `month`, `note`
`misc`	keine	`author`, `title`, `howpublished`, `month`, `year`
`phdthesis`	`author`, `title`, `school`, `year`	`type`, `address`, `month`, `note`
`proceedings`	`title`, `year`	`editor`, `volume` oder `number`, `organization`, `series`, `note`, `address`, `month`, `publisher`
`techreport`	`author`, `title`, `institution`, `year`	`type`, `number`, `address`, `month`, `note`
`unpublished`	`author`, `title`, `note`	`month`, `year`

Detailierte Informationen zu BIBTEX finden sich in [14], [10] und [12].

5.1.5

Index

Ein Index schlüsselt wichtige Begriffe eines Textes nach Seitenzahlen auf. Was ein „wichtiger Begriff“ ist, muß der Autor entscheiden.

- Für die Erstellung eines Index stellt LaTeX die *theindex*-Umgebung bereit:

  ```
  \begin{theindex}
          ⋮
  ```
  Index-Eintragungen
  ```
          ⋮
  \end{theindex}
  ```

- *Index-Eintragungen* können mit den Befehlen

  `\item` `\subitem` `\subsubitem`

  gegliedert werden und mit

  `\indexspace`

  kann Durchschuß zwischen die Indexeinträge eingebracht werden.

  Beispiel:

  ```
  \begin{theindex}
  \item Quotientenring  \dotfill 5
  \subitem nach Martindale \dotfill 6
  \subitem symmetrischer \dotfill 8
  \subsubitem beschr"ankter \dotfill 9
  \indexspace
  \item Der zentrale Tr"ager \dotfill 11
  \end{theindex}
  ```

  ergibt:

  # Index

  Quotientenring 5
  nach Martindale 6
  symmetrischer 8
  beschränkter 9

  Der zentrale Träger 11

Indexeinträge mit *MakeIndex*

Die Seitenzahl eines Begriffes, der in den Index aufgenommen werden soll, kann man von LaTeX ermitteln lassen.

Dazu gibt es zwei Möglichkeiten:

- Indexformatierung von Hand: Im Quelltext ist dazu hinter dem aufzunehmenden Begriff (ohne Leerzeichen!) der Eintrag

  `\index{`*Index_Eintrag*`}`

  vorzunehmen, wobei *Index_Eintrag* den Text bezeichnet, der in das Inhaltsverzeichnis eingetragen werden soll.
  Beispiel:

  ```
  seltene blaue Gnus\item{Gnu}
  ```

  – Trägt man `\makeindex` in die Präambel des Quelltextes (`foo.tex`) ein, so erstellt LaTeX eine Datei `foo.idx`, in der die Indexeinträge und die zugehörigen Seitenzahlen stehen.

  – Die Datei `foo.idx` wird von Hand mit der *theindex* Umgebung nachbearbeitet.

- Man läßt sich die Indexeinträge vom *MakeIndex* Programm formatieren:

  – *MakeIndex* verwendet die gleichen `\index`-Befehle wie LaTeX. Diese haben jedoch eine erweiterte Syntax; zum Beispiel lassen sich `\subitem`s im `\index`-Befehl markieren:

  ```
  seltene blaue Gnus\index{Gnu!blaues}
  ```

  ist gleichwertig zu `\item Gnu`, `\subitem blaues` in der *theindex*-Umgebung.

  – In der Präambel der Quelle `foo.tex` muß der `\makeindex`-Befehl eingetragen und das *makeidx*-Paket mit dem `\usepackage`-Befehl geladen werden.

  – Mit `makeindex foo` wird ein Index erzeugt, der durch den `\printindex`-Befehl in `foo.tex` an dessen Stelle gesetzt wird.

Ablauf von *MakeIndex*

MakeIndex ist ein universeller Indexgenerator, der auf verschiedenen Betriebssytemen (u.a. UNIX und DOS) zur Verfügung steht. Das besondere an *MakeIndex* ist seine eigene Indexsyntax, die auch den Satz komplexer Indizes erlaubt.

Indexsyntax. Es kann u.a. folgende Indexsyntax verwendet werden:

<table>
<tr><th>Symb.</th><th>Bedeutung</th></tr>
<tr><td><code>!</code></td><td>Unterteilt Subindizes (bis zu drei Ebenen) Beispiel: <code>\index{gnu!farbiges!blaues}</code></td></tr>
<tr><td><code>|(</code></td><td>Kennzeichnet den Anfang eines Seitenbereichs, auf den im Index verwiesen werden soll. Beispiel: <code>\index{gnu|(}</code></td></tr>
<tr><td><code>|)</code></td><td>Kennzeichnet das Ende eines Seitenbereichs, auf den im Index verwiesen werden soll. Beispiel: steht <code>\index{gnu|(}</code> auf Seite 5 und <code>\index{gnu|)}</code> auf Seite 11, so verweist MakeIndex auf Seite 5 – 11.</td></tr>
<tr><td><code>@</code></td><td>Wird ein Eintrag in der Form <code>\index{w@Alpha}</code> vorgenommen, so wird im Index zwar Alpha als Eintrag gedruckt, dieser aber lexikographisch unter „w“ eingeordnet.</td></tr>
</table>

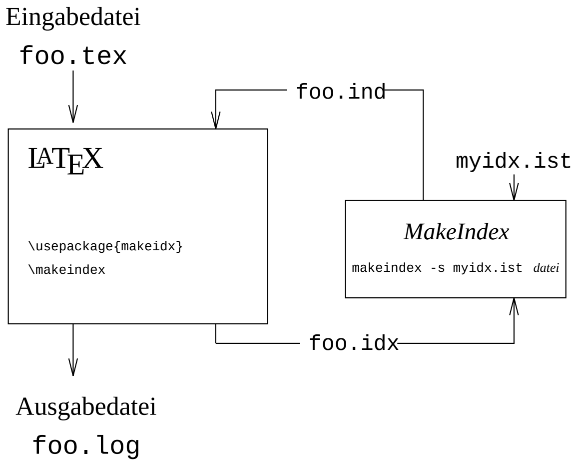
Für die Einträge in das Argument des `\index` Befehls sind folgende Einschränkungen zu beachten:

1. Befehle innerhalb von Einträgen werden erst beim Satz der `.idx`-Datei expandiert, nicht bei deren Erzeugung.

2. Der `\index`-Befehl darf keine weiteren Befehle enthalten, wenn er selbst Argument eines anderen Befehls ist (z.B. innerhalb einer Fußnote). In diesem Fall können die Befehle in der `@`-Form geschrieben werden. Beispiel: `\footnote{\index{blue@\em blue}}`.

3. Sollen Indexsyntaxbefehle wie `!`, `@` oder `|` in einem Indexeintrag gedruckt werden, so sind diese zu maskieren, d.h. durch `"!`, `"@` und `"|` zu ersetzen. Beispiel: `\index{"@-Befehl}`

Ablauf. Die Erstellung eines Index erfordert drei Schritte:



1. Wird bei einem ersten LaTeX-Lauf der `\makeindex` Befehl in der Präambel der Quelldatei `foo.tex` gefunden, so erzeugt LaTeX eine Datei `foo.idx`. In dieser `.idx`-Datei sind die mit `\index` ausgezeichneten Begriffe aus `foo.tex` den zugehörigen Seitenzahlen gegenübergestellt.

2. Bei der Bearbeitung der Datei `foo.idx` mit *MakeIndex* werden die Indexeinträge mit der *theindex* Umgebung formatiert und in der Datei `foo.ind` abgelegt.

3. Bei einem nachfolgenden LaTeX-Lauf veranlaßt der Befehl `\printindex` LaTeX dazu, eine Datei mit Namen `foo.ind` zu suchen und in den Quelltext aufzunehmen (wie `\input`).

Das *MakeIndex* Programm kann den Stil des Index mit Hilfe von Stylefiles verändern. In unserem Bild ist dies `myidx.ist`.

Für genauere Informationen siehe [11].

5.2 Graphik

5.2.1

Allgemeines

- Die Einbindung von Bildern gehört nur indirekt zur Aufgabe der Seitenformatierung (des Textsatzes).

  In der Regel besteht die Aufgabe darin, die Absätze so zu gestalten, daß Bildmaterial in die freien Flächen eingefügt werden kann.

  Zu diesem Zweck gibt es verschiedene LaTeX-Umgebungen, die entsprechende Positionierungen automatisch durchführen:

  – *figure*- und *table*-Umgebung.
  – *picinpar*-Paket.

- Ferner stehen LaTeX-Umgebungen zur Verfügung, mit denen direkt in LaTeX Graphiken erstellt werden können:

  – *picture*-Umgebung.
  – *epic*- und *eepic*-Paket.

- Komplexe, hochwertige Abbildungen sollten allerdings besser mit geeigneten, externen Graphikprogrammen erstellt werden. Die mit einem solchen Graphikprogramm erstellten und abgespeicherten Graphiken müssen dann in das LaTeX-Dokument integriert werden. Leider gab es hier in den Kindertagen von TeX/LaTeX weder Standards noch Empfehlungen, wie ein solcher zu erreichen sei. In der Folge entstanden eine Fülle unterschiedlichster Methoden, um Graphiken in LaTeX einzubinden. Inzwischen steht LaTeX zwar ein „Standardpaket“ zum Zwecke der Graphikeinbindung zur Seite, doch besitzt dieses bisher nicht das Potential, die anderen etablierten Methoden zu verdrängen. Wir besprechen kurz:

  – *bm2font*
  – *epsf*-Paket
  – *graphics*-Paket

Floating Bodies

- Die LaTeX-Umgebungen *figure* und *table* setzen Text — oder auch einfach mittels `\vspace` freigehaltenen Platz, etwa zum Einkleben eines Bildes — automatisch an eine Stelle, wo er vollständig hinpaßt, ohne durch einen Seitenumbruch zerrissen zu werden.

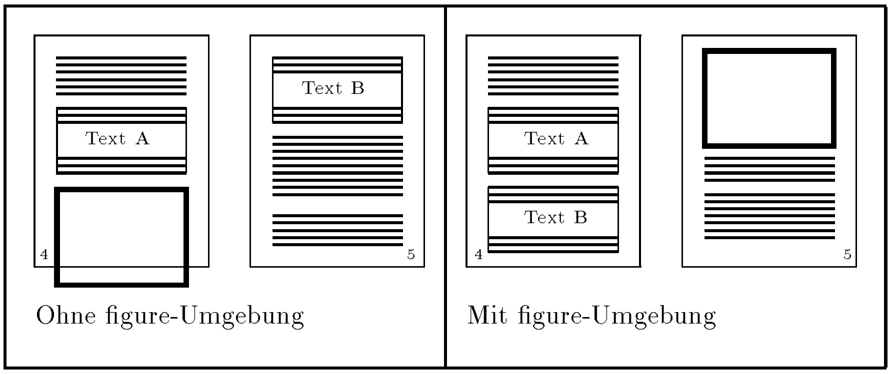


Abbildung 5.1: Floating Body

- Die allgemeine Syntax lautet:

  `\begin{`*Umgebungsname*`}[`*loc*`]`

  :

  `\caption[`*Verzeichniseintrag*`]{`*Bildunterschrift* `\label{`*Labelname*`}}`

  `\end{`*Umgebungsname*`}`

- Ein von einer solchen Umgebung eingeschlossener Bereich bildet einen sogenannten „Floating Body" (FB). LaTeX wird FBs nicht immer so plazieren, wie vom Autor gewünscht, deshalb kann ein optionales Argument *loc* angegeben werden, das die Plazierung beeinflußt. *loc* kann eines der Zeichen `h`, `t`, `b`, `p` oder `!` (oder eine Folge davon) sein; es bedeutet:

loc	Bedeutung
h	FB an exakt die Stelle des Befehls; wenn nicht möglich, dann ...
t	FB an den Anfang der nächsten Seite;
b	FB an den Fuß der nächsten Seite;
p	FBs auf einer „floatpage" sammeln.
!	Alle Restriktionen bezüglich Bildanteil aufheben.

Die `!`-Option dient dazu, beliebig viele und beliebig große Bilder auf einer Seite unterbringen zu können, da LaTeX dabei normalerweise Grenzen setzt.

Floating Bodies II

- Mit dem `\caption`-Befehl läßt sich eine Bildunterschrift angeben; Bildunterschriften lassen sich von LaTeX zu einem Verzeichnis zusammenstellen; unter anderem deshalb gibt es *zwei* FB-Umgebungen, *figure* und *table*, um separate Abbildungs- und Tabellenverzeichnisse generieren zu können.

- Innerhalb des `\caption`-Befehls läßt sich ein `\label`-Befehl benutzen, wodurch ein FB mittels `\ref`-Befehl referenzierbar gemacht wird.

- Bei Verwendung der Dokumentklassenoption *twocolumn* erzeugen die Umgebungen *figure* bzw. *table* spaltenübergreifende FBs. FBs, die „zweispaltig fließen", lassen sich mit der *figure*∗- bzw. *table*∗-Umgebung anlegen.

- Die wichtigsten Plazierungsregeln von LaTeX sind:
  - Ein FB wird so früh wie möglich gesetzt. (`h` geht allerding vor `t`).
  - Ein FB erscheint frühestens auf des Seite des Umgebungsaufrufs.
  - Bilder erscheinen nie vor früheren Bildern; entsprechendes für Tabellen.
  - Ein FB wird gemäß dem `loc`-Argument gesetzt, Voreinstellung ist `tbp`.
  - FBs können keine übervolle Seite erzeugen.

- Der Befehl `\suppressfloats[`*opt*`]` verhindert das Erscheinen weiterer FBs auf der betreffenden Seite; *opt* kann sein:
  `t`: keine weiteren FBs oben – `b`: keine weiteren FBs unten auf der Seite.

Stilparameter	Beschreibung
`\textfraction`	Minimaler Seitenbruchteil für Text.
`\topfraction`	Maximaler Seitenbruchteil für FBs oben.
`\bottomfraction`	Maximaler Seitenbruchteil für FBs unten.
`\floatpagefraction`	Minimaler Seitenbruchteil einer floatpage für FBs.
`\floatsep`	Vertikaler Zwischenraum zwischen FBs.
`\textfloatsep`	Vertikaler Zwischenraum zwischen FBs und Text.
`\intextsep`	Vertikaler Zwischenraum zwischen FBs und Text bei Verwendung der `h`-Option.
`topnumber`	Maximale Anzahl von FBs oben.
`bottomnumber`	Maximale Anzahl von FBs unten.
`totalnumber`	Maximale Anzahl von FBs auf einer Seite.

picinpar-Paket

- Das *picinpar*-Paket stellt die drei Umgebungen *window*, *figwindow* und *tabwindow* zur Verfügung, mit deren Hilfe sich Text, ein Bild oder eine Tabelle in ein rechteckiges Fenster setzen läßt, welches sich inmitten oder am Rande eines Absatzes befinden darf.

- Die allgemeine Syntax lautet:
  `\begin{`*U-Name*`}[`*Überhang*`,`*pos*`,{`*F-Inhalt*`},{`*F-Unterschrift*`}]`
  Absatztext ...
  `\end{`*U-Name*`}`

Parameter	Beschreibung
U-Name	*window*, *figwindow* oder *tabwindow*.
Überhang	Anzahl der Zeilen über der Fensteroberkante.
pos	`l`, `c` oder `r`, je nachdem ob das Fenster am linken Rand, mittig oder am rechten Rand positioniert werden soll.
F-Inhalt	Bei *window* ein Text, bei *figwindow* eine Abbildung oder bei *tabwindow* eine Tabelle.
F-Unterschrift	Wirkt bei *figwindow* und *tabwindow* analog dem `\caption`-Argument bei *figure* bzw. *table*.
Absatztext	Der komplette Absatz, in den das Fenster eingebaut werden soll.

Beispiel:

```
\begin{window}[2,c,{\myfont V},{\small Das Siegeszeichen}]
In einer bekannten amerikanischen Science-Fiction Serie...
\end{window}
```

In einer bekannten amerikanischen Science-Fiction Serie, die nach dem Strickmuster von Dallas und ähnlichen Seifenopern gedreht wurde, fühlt sich ein Überlebender des Holocaust lebhaft an das Dritte Reich erinnert, als Außerirdische, die angeblich aus bloßem Interesse an fairem Handel auf die Erde gekommen sind und eine Art Wirtschaftshilfe zu leisten beginnen, eines Tages die ökonomischen Schwierigkeiten der Menschheit den Wissenschaftlern anlasten und Progrome gegen dieselben starten. Dieser Überlebende sprüht das erste rote „V“ an eine Wand, welches schon bald zum Zeichen des Widerstandes gegen die als Menschen maskierten außerirdischen Echsen wird ...

V

Das Siegeszeichen

picture-Umgebung

- Die *picture*-Umgebung stellt stellt mit dem `\put`-Befehl ein Werkzeug zur Verfügung, um graphische oder Textelemente frei zu positionieren; d.h. die Elemente dürfen auch außerhalb der Bildabmessungen liegen. Bei der Positionierung verwendet die *picture*-Umgebung eine interne Einheit, die mit dem Befehl `\setlength{\unitlength}{`*unit*`}` gesetzt werden kann.

- Die allgemeine Syntax lautet:

  `\setlength{\unitlength}{`*unit*`}`
  `\begin{picture}(`*width*`,`*height*`)(`*x-origin*`,`*y-origin*`)`
  `\put(`*x*`,`*y*`){`*picture*-Befehl`}`
  :
  `\end{picture}`

Parameter	Beschreibung
unit	Eine Bildeinheit entspricht dem hier angegebenen Maß.
width	Breite des Bildes in Bildeinheiten.
height	Höhe des Bildes in Bildeinheiten.
x-origin	X-Koordinate des Bildursprungs in Bildeinheiten.
y-origin	Y-Koordinate des Bildursprungs in Bildeinheiten.
x	X-Koordinate des zu plazierenden Elementes.
y	Y-Koordinate des zu plazierenden Elementes.

- *picture*-Befehle können sein:

Befehl	Beschreibung
`\line(`*a,b*`){`*lngth*`}`	Linie der Steigung *a/b* und der Länge *lngth*.
`\vector(`*a,b*`){`*lngth*`}`	Vektor der Steigung *a/b* und der Länge *lngth*.
`\circle{`*diam*`}`	Leerer Kreis des Durchmessers *diam*.
`\circle*{`*diam*`}`	Gefüllter Kreis des Durchmessers *diam*.
`\oval(`*wdth,hght*`)`	Einem Rechteck der Breite *wdth* und der Höhe *hght* einbeschriebenes Oval.
`\makebox`	Kurzer Text / einzelne Zeilen; siehe Folie 3.2.6.
`\framebox`	Eingerahmter Text; siehe Folie 3.2.6.
`\shortstack`	Mehrere, ausgerichtete Zeilen; siehe [12].
`\parbox`	Vollständige Absätze; siehe Folie 3.2.7.

Mehr zur *picture*-Umgebung

Anmerkungen:

In der *picture*-Umgebung stehen zwei verschiedene Linienstärken für die graphischen Bildelemente zur Verfügung. Zwischen diesen kann mit den Befehlen `\thicklines` und `\thinlines` umgeschaltet werden. Voreingestellt ist `\thinlines`.

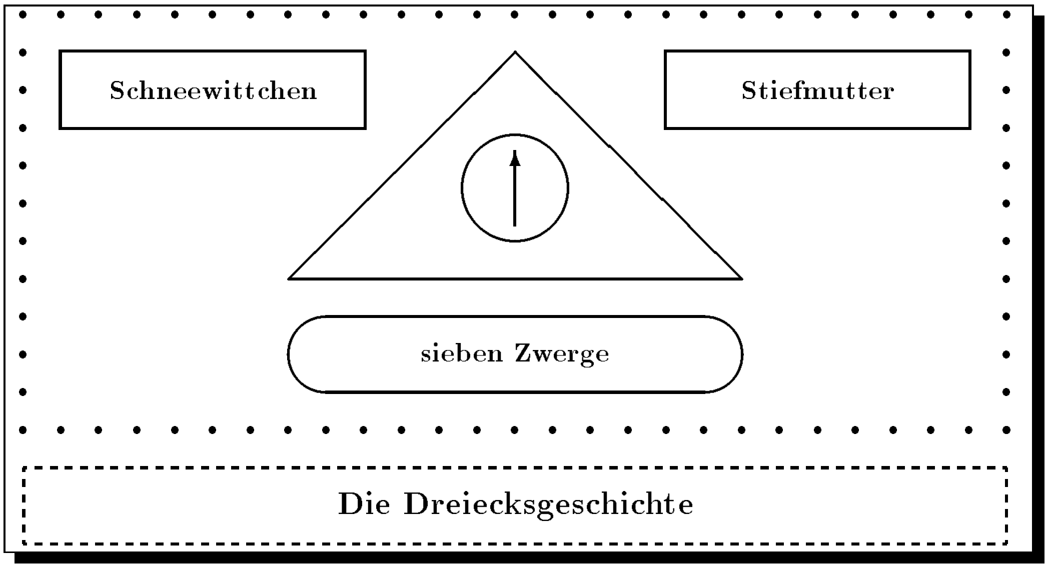
Außer dem Positionierungsbefehl `\put` gibt es auch noch den Positionierungsbefehl `\multiput`. Dieser hat zwei Inkrementargumente für die X- und Y-Koordinate und einen Wiederholungsfaktor zusätzlich; siehe Beispiel.

Außer dem Befehl `\framebox` gibt es auch noch die Befehle `\dashbox`, `\fbox` und `\frame`. Für deren Syntax vergleiche [12].

Auch lassen sich in der *picture*-Umgebung Bezier-Kurven und Gitter zeichnen, hierfür sei ebenfalls auf [12] verwiesen.

Beispiel:

```
\setlength{\unitlength}{1mm} \thicklines
\begin{picture}(130,70)
\put(0,0){\dashbox(130,10){{\large\bf Die Dreiecksgeschichte}}}
\multiput(0,15)(5,0){27}{\circle*{1}}
\multiput(0,70)(5,0){27}{\circle*{1}}
\multiput(0,15)(0,5){11}{\circle*{1}}
\multiput(130,15)(0,5){11}{\circle*{1}}
\put(65,20){\makebox(0,10){{\normalsize\bf sieben Zwerge}}}
\put(35,35){\line(1,0){60}}
\put(35,35){\line(1,1){30}}
\put(95,35){\line(-1,1){30}}
\put(65,47){\circle{15}} \put(65,25){\oval(60,10)}
\put(65,42){\vector(0,1){10}}
\put(5,55){\framebox(40,10){{\normalsize\bf Schneewittchen}}}
\put(85,55){\framebox(40,10){{\normalsize\bf Stiefmutter}}}
\end{picture}
```



epic- und eepic-Paket

- Die *picture*-Umgebung hat viele Schwächen, von denen ein Großteil durch Verwendung des *epic*-Pakets (enhanced picture-environment) entfällt:
  - Die *picture*-Umgebung erlaubt nur bestimmte Liniensteigungen, da *epic* Linien aus Teillinien zusammensetzen kann, sind beliebige Steigungen möglich, auch wenn die Linien nicht immer schön aussehen.
  - *epic* kennt Linienzüge, gestrichelte und gepunktete Linien.
  - Der `\multiput`-Befehl wurde im *epic*-Paket erweitert.
  - *epic* verfügt über einen eigenen Befehl zum Zeichnen von Gittern/Rastern.

- Nicht gelöst wurde im *epic*-Paket das Problem der Kreisradien, von denen LaTeX nur 22 kennt. Neue Befehle / Umgebungen des *epic*-Pakets:

Befehl	Beschreibung
`\drawline`	Verbessertes `\line`.
`\multiputlist`	Erweitertes `\multiput`.
`\matrixput`	Weitere `\multiput`-Variante.
`\grid`	Neuer Befehl zum Zeichnen von Rastern.
`\dottedline`	Gepunktete Linien.
`\dashline`	Gestrichelte Linien.
`\jput`	Für Linienzüge.

Umgebung	Beschreibung
`\join`	Für durchgezogene Linienzüge.
`\dashjoin`	Für gestrichelte Linienzüge.
`\dottedjoin`	Für gepunktete Linienzüge.

- Das *eepic*-Paket erweitert die *picture*-Umgebung nochmals. Mit *eepic* lassen sich nun endlich auch beliebige Kreisradien zeichnen, alledings greift *eepic* zu diesem Zweck auf `\special`-Befehle zurück, die nicht mit allen dvi-Treibern zusammenarbeiten. Aus diesem Grunde ist *eepic* nur eingeschränkt zu empfehlen. Einen Teil der *eepic*-Befehle kann man dvi-Treiber-unabhängig mit dem Paket *eepicemu* emulieren, dies hat aber den Nachteil, das es in TeX erhebliche Rechenzeit beansprucht.

- Für die Syntax der *epic*- und *eepic*-Befehle sei auf [17] verwiesen.

Einbindung von Rastergraphiken

Während die LaTeX-*picture*-Umgebung oder z.B. das *epic*-Paket quasi die Darstellung von *Vektorgraphiken* ermöglichen[1], wird hier besprochen, wie *Rastergraphiken*, z.B. eingescannte Photographien, in LaTeX eingebunden werden können. Drei Verfahren haben sich hier etabliert:

- *bm2font* (bitmap to font) von Friedhelm Sowa macht aus einer Raster- (BitMap-) Graphik einen von TeX nutzbaren Font und generiert automatisch die nötigen TeX- und LaTeX-Befehle, um das zerstückelte Bild in TeX/LaTeX wieder zusammenzusetzen, sprich einzubinden. Die Methode ist zwar vergleichsweise kompliziert, jedoch sehr durchdacht und hat den entscheidenden Vorteil, völlig unabhängig vom verwendeten dvi-Treiber zu sein. Nähere Informationen finden sich in [17] oder [9].

- Das *epsf*-Paket von Tomas Rokicki braucht zur Ausgabe der Graphik einen dvi-Treiber, der die `\special`-Befehle des Pakets versteht, z.B. den ebenfalls von Rokicki stammenden und sehr verbreiteten Treiber `dvips`.

  Die einfachste Methode, das Paket zu nutzen, besteht darin, einen Aufruf folgender Bauart in ein LaTeX-Dokument einzufügen:

  ```
  \begin{center}
    \epsffile{filename.eps}
  \end{center}
  ```

  Das Paket handhabt eigentlich ausschließlich „Encapsulated-PostScript-Files", die BitMaps enthalten dürfen, kann unter Umständen aber auch mit einem normalen PostScript-File fertigwerden, sofern es ihm gelingt, die Abmessungen des Bildes auszurechnen. Zu Details siehe [17].

- Das zu LaTeX 2_ε gehörige *graphics*-Paket ermöglicht eine Rastergraphik-Einbindung ähnlich der des *epsf*-Pakets, indem es für vom Treiber unterstützte Dateitypen `\special`-Befehle generiert. Wird z.B. `dvips` benutzt, kann das *graphics*-Paket ebenfalls PostScript-Files einbinden:

  ```
  \begin{center}
    \includegraphics{filename}
  \end{center}
  ```

  Ein optionales Argument ermöglicht beim `\includegraphics`-Befehl die Wahl eines Bildausschnitts. Für die genaue Syntax siehe [12].

[1] Was einige Graphikprogramme ausnutzen, indem sie Verfahren zur Verfügung stellen, um selbst sehr komplexe Vektorgraphiken z.B. als LaTeX-*picture*-Befehlsdatei zu exportieren.

5.3 Bearbeitung langer Dokumente

5.3.1

Dokumentuntergliederung

Ein längeres Dokument gliedert sich in aller Regel in viele verschiedene Bestandteile auf, vom Titel bis zu Index und Glossar. LaTeX stellt Umgebungen und Werkzeuge bereit um

- diese Textteile zu formatieren und auf diese Teile geeignet zu verweisen (LaTeX-Konstrukte),
- sowohl die Teile als auch die Verweise übersichtlich zu verwalten (Tools).

Soweit solche Konstrukte/Tools in diesem Buch beschrieben sind, verweist die folgende Tabelle auf die entsprechende Stelle.

Bestandteil	LaTeX-Konstrukt	Siehe	Tool
Titel	`\maketitle`, *titlepage*-Umgebung	5.3.2	–
Zusammenfassung	*abstract*-Umgebung	5.3.2	–
Inhaltsverzeichnis	`\tableofcontents`	2.3.7	–
Tabellenverz.	`\listoftable`	2.3.7	–
Abbildungsverz.	`\listoffigures`	2.3.7	–
Textgliederung	Gliederungsbefehle: `\part`, `\chapter`, `\section`, ...	2.3.2	5.3.6
	Laden einzelner Abschnitte mit `\input` oder `\include`	5.3.3	5.3.5
Anhang	*appendix*-Umgebung	–	–
Literaturverz.	*bibliography*-Umgebung; BibTeX	5.1.1	5.3.7
Index und Glossar	*theindex*-Umgebung, `\makeglossary`, `\makeindex`; *makeidx*-Paket, *MakeIndex* Programm	5.1.5	5.3.7

5.3.2

Titelseite und Zusammenfassung

Eine Titelseite kann mit dem `\maketitle` Befehl automatisch formatiert werden. Der Titel erscheint bei den Dokumentklassen *report* und *book* auf einer eigenen Seite ohne Seitennummer, bei *article* wie im unteren Beispiel.

- `\title` und `\author` müssen gesetzt werden. Der Titel ist mittels `\maketitle` abzuschließen.
- Ein Datum wird automatisch erzeugt, wenn `\date` nicht gesetzt wird.
- Der Befehl `\thanks{...}` entspricht dem `footnote`-Befehl.

Eine Zusammenfassung kann in der *abstract* Umgebung mit Randeinzug und kleiner Schrift erzeugt werden.

Beispiel:

```
\documentclass[11pt,german,twoside]{article}
\usepackage{babel}

\begin{document}
\title{{\bfseries Textsatz und Layout mit \LaTeX}}
\author{\textsc{Ralf Banning} und \textsc{Heiko A.~Groeneveld}}
\date{1995}
\maketitle

\begin{abstract}
    \LaTeX{} ist ein von Leslie Lamport entwickeltes ...
\end{abstract}

Als universelles Satzsystem entwickelt, er"offnet ...

\end{document}
```

> **Textsatz und Layout mit LaTeX**
>
> Ralf Banning und Heiko A. Groeneveld
>
> 1995
>
> **Zusammenfassung**
>
> LaTeX ist ein von Leslie Lamport entwickeltes Makropaket für TeX, das Layoutbefehle bereitstellt, die sich an der logischen Struktur von Textdokumenten orientieren.
>
> Als universelles Satzsystem entwickelt, eröffnet TeX alle Möglichkeiten des Textsatzes, freilich mit dem Nachteil, daß der Benutzer einem ungewöhnlich großen Sprachumfang von etwa 900 Befehlen gegenübersteht ...

5.3.3

Strukturierte Quelldateien

Lange LaTeX-Quelltexte werden erheblich übersichtlicher, wenn sie in mehrere Dateien zerlegt werden. Dies wird durch die folgenden Befehle unterstützt:

- `\input{`*name*`}` liest den Inhalt der Datei *name* ein und setzt ihn an die Stelle im Quelltext, wo der `\input`-Befehl steht.
- `\include{`*name*`}` wirkt wie `\input`, bewahrt aber die Referenzierung ausgewählter Dateien; `include` beginnt immer eine neue Seite!

Beispiel: Eine Diplomarbeit über „Metatheorien" (Datei „`meta.tex`"):

Alte Quelldatei `meta.tex`	Datei Struktur	Neue Quelldatei
`\title{Metatheorien}` `\author{U.N.\ Known}` `\maketitle`	`title.tex`	`\input{title}`
`\tableofcontents`	`meta.lot`	`\tableofcontents`
`\chapter*{Vorwort}` *Das Vorwort*	`vorwort.tex`	`\input{vorwort}`
`\chapter{Grundlagen}` *Text des 1. Kapitels*	`kap1.tex`	`\input{kap1}`
`\chapter{Das Theorem}` *Text des 2. Kapitels*	`kap2.tex`	`\input{kap2}`
`\chapter{Anwendungen}` *Text des 3. Kapitels*	`kap3.tex`	`\input{kap3}`
`\bibliographystyle{plain}` `\bibliography{diplom}`	`lit.tex`	`\input{lit}`

Auswahl von Textteilen und Hierarchien

Vor allem bei langen Texten ist es nicht effektiv, jedesmal die ganze Quelldatei mit LaTeX zu „compilieren“, wenn etwa nur ein einzelnes Kapitel gerade bearbeitet wird. Strukturierte Quelldateien erlauben eine Teilbearbeitung.

Dabei unterscheiden sich zwei Vorgehensweisen:

- Werden die Teildateien mit `\input` gelesen, kommentiert man alle `\input`-Zeilen aus, deren Dateien nicht bearbeitet werden sollen.
- Werden die Teildateien mit `\include` gelesen, so kann man mit der Zeile `\includeonly{`*nam1,nam2, ...*`}` die `\include`-Dateien auswählen, die man bearbeiten möchte. Dieser Befehl muß in der Präambel stehen.

Beispiel:

Bei einer Bearbeitung von

```
\documentclass{report}
\includeonly{vorwort,kap1}
\begin{document}
\input{titel}
%\tableofcontents
\include{vorwort}
\include{kap1}
\include{kap2}
\include{kap3}
%\input{lit}
\end{document}
```

werden nur die Dateien `titel.tex`, `vorwort.tex` und `kap1.tex` berücksichtigt.

Der `\include`-Befehl legt für jede von ihm eingelesene Datei eine eigene `.aux`-Datei an. Wählt man daher die zu bearbeitenden Textteile mit `\includeonly` aus, bleiben bisher erstellte Kreuzreferenzen erhalten (im Gegensatz zur `\input`-Auswahl!).

Änderungen im Text können allerdings im nachfolgenden Text zu falschen Seitenreferenzen führen, die sich nur durch vollständiges Übersetzen aller Quelldateien beseitigen lassen.

Kontrolle von Dateihierarchien

Sind die Teildateien eines zerlegten Quelltextes immer noch zu lang oder zu unübersichtlich, können die Teildateien ihrerseits wieder zerlegt werden:

- Eine Datei, die mit `\input` geladen wird, kann ihrerseits Dateien mittels `\input` laden.
- Eine Datei, die mit `\include` geladen wird, kann ihrerseits Dateien mit `\input` laden, aber *nicht* mit `\include`.

Beispiel:

Die Datei `kap1.tex` wird in einzelne Abschnitte `sec1_1.tex`, `sec1_2.tex`, ... zerlegt und könnte dann so aussehen:

Datei `kap1.tex`:

```
\chapter{Grundlagen}
\input{sec1_1}
\input{sec1_2}
\input{sec1_2}
```

Um den Überblick über die Teildateien eines Quelltextes zu behalten, kann in der Präambel der Befehl `\listfiles` verwendet werden. Dadurch wird eine Liste der aktuell verwendeten Dateien auf dem Bildschirm und in der `.log`-Datei ausgegeben.

Beispiel:

Mit `\includeonly{vorwort,kap1}` ergibt sich die folgende Liste:

```
 report.cls  1994/12/09 v1.2x Standard LaTeX document class
 size10.clo  1994/12/09 v1.2x Standard LaTeX file (size option)
vorwort.tex
   kap1.tex
sec1_1.tex
sec1_2.tex
sec1_2.tex
***********
```

5.3.6

Verwaltung von Kreuzreferenzen

Die in den `\label`-Befehlen verwendeten Marken sind in der Ausgabe nicht mehr sichtbar. Folgende Werkzeuge geben Aufschluß über die verwendeten Marken und deren Position.

- Die Datei `lablst.tex` liest bei ihrer Bearbeitung eine frei wählbare LaTeX-Quelldatei ein. Daraus erstellt sie ein Verzeichnis aller verwendeten Marken, dem Zählerstand und der Seitenposition der Marke. Am Ende des Verzeichnisses folgt eine Aufzählung der Bezugsmarken der Bibliographie.

  Beispiel:

  1 Programmstruktur von LaTeX **6**

  1.1 Was ist Textsatz? 9

  1.1.1 Was ist LaTeX? 11

  1.1.2 Lettern unter sich 15

  `latex` 1.1.1 Page: 15

  `lettr` 1.1.3 Page: 22

  1.1.3 Der Zeilenumbruch 23

  `Knuth3` [2]

  `B&W931` [1]

- Das Paket *showkeys* stellt die im `\label`-Befehl verwendeten Marken auf dem Seitenrand in einem Rahmen dar. Zusätzlich werden auch für alle `\ref`, `\pageref` und `\cite`-Befehle die Marken dargestellt, auf die sie verweisen. Diese Marken werden hierbei in der Ausgabe über der Referenz dargestellt.

  Beispiel:

```
\section{Querverweise}\label{Marke}

Ein Querverweis (siehe Abschnitt \ref{Marke}) wird ...
```

11.9 Querverweise `Marke`

Ein Querverweis (siehe Abschnitt `Marke` 11.9) wird ...

5.3.7

Verwaltung von Indizes und Bibliographiemarken

Mit dem Paket *showidx* werden auf jeder Seite die mit `\index` vorgenommenen Eintragungen am Seitenrand wiedergegeben.

Beispiel:

```
Gute Rotweine\index{Rotwein} findet man auch im
Burgund\index{Burgund@\textit{Burgund}}.
```

Gute Rotweine findet man auch im Burgund. Rotwein *Burgund*

Wird das Paket *showtags* verwendet, so werden die Bezugsmarken der Bibliographie im Literaturverzeichnis mit abgedruckt.

Beispiel:

```
\begin{thebibliography}{XXX}
\bibitem[Lam]{Lam94} L.~Lamport: {\itshape \LaTeX,
  A Document Preparation System, User's Guide and Reference
  Manual}, 2nd ed, Addison-Wesley Publishing Company (1994)
\bibitem[Kop]{Kop94} H.~Kopka {\itshape \LaTeX, Bd.\ 1.\
  Einf"uhrung}, Addison-Wesley Publishing Company (1994)
\end{thebibliography}
```

Literaturverzeichnis

Lam94

[Lam] L. Lamport: *LaTeX, A Document Preparation System, User's Guide and Reference Manual, 2nd ed, Addison-Wesley Publishing Company (1994)*

Kop94

[Kop] H. Kopka *LaTeX, Bd. 1. Einführung*, Addison-Wesley Publishing Company (1994)

Literaturverzeichnis

[1] R. Banning and H. A. Groeneveld. *LaTeX — ein Formatierungsprogramm für Dokumente*. Schriftenreihe des ZDV Nr. 1. Zentrum für Datenverarbeitung Universität Tübingen, 1995. 4. überarbeitete Auflage.

[2] Ralf Banning. LaTeX 2ε: Eine neue LaTeX-Generation am ZDV. *Benutzerinformation des ZDV der Universität Tübingen*, 1+2:1–5, 1995.

[3] U. Baufeldt, M. Dorra, H. Rösner, and J. Scheuermann. *Informationen übertragen und Drucken*. Grundfachkunde Druckindustrie. Verlag Beruf + Schule, Elmshorn, 1977.

[4] M. Goosens, F. Mittelbach, and A. Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1994.

[5] Kurt Kirchner. *Satz, Druck, Einband und verwandte Dinge*. F. A. Brockhau, Wiesbaden, 1970.

[6] D.E. Knuth. *The TeXbook*. Addison Wesley, 1986. 17. überarbeitete Auflage 1990.

[7] D.E. Knuth. *The METAFONT book*. Addison Wesley, 1991.

[8] Donald E. Knuth. Typesetting concrete mathematics. *TUGboat*, 10:31 – 36, 1989.

[9] H. Kopka. *LaTeX-Erweiterungsmöglichkeiten*. Addison Wesley, 1992. 3. überarbeitete Auflage.

[10] H. Kopka. *LaTeX*. Bd. 1 Einführung. Addison Wesley, 1994. 1. Auflage 1994.

[11] Leslie Lamport. *MakeIndex: An Index Processor For LaTeX*, 17 February 1987.

[12] Leslie Lamport. *LaTeX: A Document Preparation System*. Addison Wesley, Massachusetts, second edition, 1994.

[13] Ralf Nebelo. Formelsatz. *c't*, 6:186–195, 1995.

[14] Oren Patashnik. BIB*TeXing*, February 8, 1988.

[15] The LaTeX 3 Project. *LaTeX 2ε for authors*, 1994.

[16] The LaTeX 3 Project. *LaTeX 2ε for class and package writers*, 1994. Preliminary draft June 1994.

[17] Friedhelm Sowa. *TeX/LaTeX und Graphik*. Springer-Verlag, 1994.

Verzeichnis der Folien und Ergänzungsseiten

Index